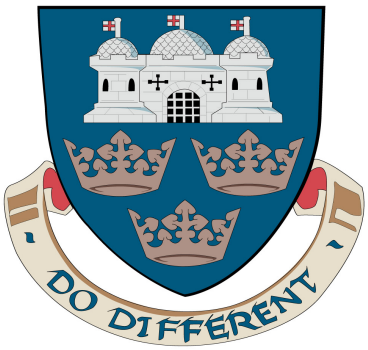
Language identification using text, audio and video feature mapping

Zhuoyi Dai

A thesis submitted for the Degree of
Doctor of Philosophy

University of East Anglia
School of Computing Sciences



July, 2018

Abstract

Unlike text language identification techniques, which are now quite mature, audio and video language identification techniques still face many challenges. One of the main challenges, due to a variety of reasons, is that there are not enough audio and video datasets.

However, text data are sufficient for experiments and many text databases are free for research which leads to an interesting question: can we identify an unknown video or audio language based on the relationship between the known text languages? To answer this question, it requires us to examine two issues: language identification and language mapping.

In language identification, we compare two methods which are zipping classification and N-gram modelling. An advantage of zipping classification is that it tolerates the lack of long training data and can be applied to a large variety of problems without modification. However, the N-gram model provides a high classification accuracy and efficiency which makes it worthy of consideration. Also, we evaluate another audio classification method based on the MPEG compression to compare with the general zipping tools and the N-gram model.

For the language mapping section, we firstly use the Robinson-Foulds tree distance to measure the distances between the language trees and also use Sammon mapping and Shepard's interpolation to map the language distance results from the higher dimensions to the lower dimensions and try to find the optimal language relationships in the specific dimension.

for my Grandpa, who has my love, as always.

Acknowledgements

My thanks to Richard Harvey for his kindness and patience. His great experienced and intelligent supervision made my PhD study period helpful. He also supplied me with many suggestions on resources for my research which have been useful.

Thank you to Yuxuan Lan, who supported my visual and audio feature tracking. Thank you to Barry Theobald and Stephen Cox for their advice on testing my results. Barry also for donating his new Mac after he left to work for Apple. Thank you to Jacob Newman for sharing his visual features for this project and patiently teaching how to compute the visual tracks.

Thank you to Christopher Powell, his expertise in the GRACE high-performance computing helped me to improve my efficiency pretty much.

Thank you to Dan Smith, Wenjia Wang, Pierre Chardaire and Ellis Kurland, who provide teaching jobs for my working experience. Thank you to my colleagues, Daivd Cutting, and Geoffrey Guile, who taught me how to explain ideas to students which helped my oral presentation skills.

My appreciation to my landlord, Zhigang Dong, who understand my difficulties and is kind to provide delicious food when I am too busy to cook. He is also glad to help me to get used to the new environment in UK. Thank you to my best friend, Qiuping Sun, as an optimistic girl, she is always happy to share goodness and happiness with me.

This dissertation is dedicated to my parents, without their love and support, I could never have had the chance to complete my PhD degree.

Contents

List of Abbreviations

Abbreviation	Meaning
AAM	Active Appearance Model
AV	Audio-Visual
ASR	Automatic Speech Recognition
CMOS	Complementary Meta-Oxide-Semiconductor
DCT	Discrete Cosine Transform
EMA	Electromagnetic Articulography
FMN	Feature Mean Normalisation
GMM	Gaussian Mixture Model
GPDF	Gaussian Probability Density Function
HD	High Definition
HMM	Hidden Markov Model
HTK	Hidden Markov Model Toolkit
IMELDA	Integrated Mel-scale representation with LDA
IPA	International Phonetic Alphabet
LDA	Linear Discriminant Analysis
LID	Language Identification
MFCC	Mel-Frequency Cepstral Coefficients
PCA	Principal Component Analysis
PDF	Probability Density Function
PDM	Point Distribution Models
PRLM	Phone Recognition followed by Language Modelling
PPRLM	Parallel Phone Recognition followed by Language Modelling
SVM	Support Vector Machine
UN	United Nations
VLID	Visual Language Identification
VQ	Vector Quantisation

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Aims

Since Mustonen [1965] used MDA (Multiple Discriminant Analysis) to solve problems in text language identification, there is a large number of researchers investigating in this area. Thus, after decades of studying text language identification, there are some techniques are well mature and well coded. Some of those techniques, such as the N-gram model, have high accuracy with great efficiency and are still being used today. One of the advantages of text language identification is that it is easy to collect text datasets thanks to the development of the Internet. Hence, researchers can build up a dataset containing multiple languages in a short time. Also, there are standard encoding sets to ensure the format of character encoding does not impact on research results.

For audio language identification, there are also mature datasets which are accepted by most researchers and are used as part of standard evaluations. Although it is impossible to collect a dataset representing all languages due to political and cultural issues, audio language datasets are still comparably larger than video datasets. In other words, there are lots of standard audio language datasets available online (such as NIST datasets), while there are only a few video datasets can be

used for language identification. Since Zissman [1996] became interested in audio identification, there have been many researchers involved in speech recognition and language identification and have been reported that they have achieved high recognition accuracy.

Although video language identification might be useful in security and business applications where audio or text is not available, it might also provide enhanced audio language identification, particularly in noisy conditions when audio identification can fail. That said, there are very few video language identification databases so the field is highly undeveloped.

This thesis aims to study the problem of audio and video language identification, based on textural language information. We wonder if the distances between languages in text, audio, and video have some relationship. If so, then the unknown audio or video language could be classified or nearly classified by comparing its relationship to other audio or video signals and for similarities in the relationship between text languages.

Chapter 2

Literature review

2.1 Introduction

This thesis is concerned with the identification of human language, in either its written form (text language identification or TLID), acoustic form (audio language identification or ALID) or from the appearance of the mouth region of the speaker (video language identification or VLID).

By applying the research of child learning, as one of the earliest papers on language identification, Gold [1967] introduced the idea of language learnability. He remarked that, for effective language learnability, it was essential to consider all aspects of language from orthography through to semantics. Gold [1967] defined language identification (LID) as 'a method to learn an unknown language using a corpus of specified languages through its presented information'. Gold's ideas, which focused on text language identification, have been widely accepted and the resulting LID technologies have been developed in three branches:

- Text Language IDentification (TLID);
- Audio Language IDentification (ALID);
- Video Language IDentification (VLID);

Table 2.1 lists the major languages of the world with estimates of the number of speakers globally.

Mandarin, English, Arabic, French, Russian and Spanish are the current official languages of the UN. Besides French, the other five languages are used widely around the world. In this report, we concentrate on Mandarin, English and Arabic for the target languages because they have many speakers and so are easier to record.

The problems with current VLID systems relate to recognition accuracy. In addition, there is no such thing as a standard video data corpus that is available and open to all researchers.

In contrast, there are large numbers of multilingual text corpora available and a commonly used audio language data corpus: NIST 22. In summary, it seems much easier to build a high-accuracy TLID or ALID system than a VLID system. Now, we ask what issues might arise if we were to use TLID or ALID to improve VLID?

The first challenge is whether the data resources are reliable. Are the data normalised and well stratified by gender, age or other factors? Does it contain other forms of unnecessary noise that would reduce accuracy? Does the format of the input affect our result?

Table 2.1: List of languages with more than 100 million native speakers. Unlike European languages, which consist of initials and finals, the size of the Chinese alphabet represents the number of pronunciations in the Pinyin. Also, the size of the Japanese alphabet represents the number of pronunciations in the Latin alphabet [Lewis and D.Fenning, 2013].

Language	**Native speaker**	**Non-native speaker**	**Phoneme size**
Mandarin	848 million	1026 million	56
Spanish	406 million	466 million	48
English	335 million	765 million	44
Hindi	260 million	380 million	$44-51$
Arabic	206 million	380 million	34
Portuguese	202 million	217 million	28
Bengali	193 million	250 million	36
Russian	162 million	272 million	45
Japanese	122 million	123 million	20

The second question is how to build a recognition system for TLID, ALID, and VLID? There are so many techniques that can be used to carry out this task, but which would perform best? Which kind of features will best fit our requirements?

The third question, which is the key question we need to address, is how to map TLID and ALID results to VLID.

We provide an introduction to these questions in the subsequent chapters.

2.2 TLID (Text Language IDentification)

Text Language Identification (TLID), which is also described as written language classification, plays a key role in machine translation and information extraction, as well as in other areas such as information retrieval. Although some papers on language identification were published before 1967, such as [Mustonen, 1965], Gold [1967] might have been the first to define language identification as a method of learning an unknown language and deciding which class to which the unknown language belongs. He also devised simple rules for TLID. Some of his assumptions, such as the assumption that the personal style of the writer was less distinctive than the language, remain commonplace. Furthermore, his observation that TLID needs well-defined information on the character sets, spelling and grammar of the language is also helpful.

A useful summary of the state-of-the-art before 1996 in TLID is given by Sibun and Reynar [1996]. To compare these language identification methods, they discussed some issues as standards for evaluation. The text-language-identification-based issues discussed below are features, language selection, algorithm, text encoding, input format, size of text and evaluation methods.

The first question mentioned by Sibun and Reynar [1996] is what kind of features might be suitable for TLID? From previous research, characters, words, and phonemes are frequently used. Other linguistic rules might also help, such as morphology, syntax, semantics, pragmatics, and graphemics. Morphology is defined as a method

of analysing words' construction and their relationships with each other based on morphemes. For example, for a word like 'happy', there are several words sharing the same element of 'happy' such as 'happily', 'unhappy' and 'happiness'. In this case, morphology is used to explain the nature of the connections between those four words. Syntax defines the rules on governing how sentences are organised. Syntax, semantics, and pragmatics are all concerned with the transmission of meaning. Syntax concentrates mainly on the relationships between words, while semantics is used for the study of s igns, such as words, and what they refer to. Pragmatics is the study of what sentences mean for users and interpreters and how that the meaning is transmitted through the context of the utterance. Graphemics is related to writing systems. Although, in most cases, the writing system is quite simple, there are some exceptions that might present a challenge for text language identification. For example, some languages might not have consistent writing, such as Arabic, which is not spoken as it is written. Some languages have no writing system and some languages might have multiple writing systems, such as Japanese, which has two types of the character set, one of which is logographic, called kanji, and originates from Chinese characters, and the other is syllabic, called kana [Collinge, 2002].

Sibun and Reynar [1996] also mentioned the importance of language selection. Different decisions on language selection might impact on identification accuracy. Based on a widely accepted linguistic method known as genetic classification, an example is given by Ruhlen [1991] is that Portuguese, Spanish, Catalan, French, Italian, Sardinian and Romanian all originated from Latin, so are more likely to be confused than, for example, Mandarin and Russian.

With the development of the internet, the number of languages that can be collected electronically has increased dramatically. However, to include a variety of languages, the early stages of text encoding were different from that which we use currently. Many historic TLID databases used ASCII (American Standard Code for Information Interchange), which is a coding highly tuned to American English but then revised for other languages as the internet is used internationally. In addition,

there are multiple encoding standards that are used in parallel today. To train and test the language identification model, it is necessary to ensure all language texts are encoded in uniform character sets.

Input formatting is concerned with whether resources come from online texts, images or other methods. Most language identification methods are based on online text but some applications work on Optical Character Recognition (OCR) [Peake and Tan, 1997; Hochberg et al., 1999]. Since we use only online text resources in this thesis, we do not discuss OCR in detail.

It is common to find that language identification accuracy will improve with the increasing size of the text. However, some languages might not have such large datasets to support long string training and testing tasks. An ideal language identification algorithm should be able to identify languages in a short period with high accuracy. Additionally, accuracy should improve as the training and testing data expands. The lack of training data can make the language identification model unable to describe the data features. Also, the lack of testing data can not indicate the complexity of the languages and would reduce the identification accuracy [Manning and Schütze, 1999].

In Section 2.2.1 to 2.2.5, we will discuss TLID based on these issues and give a brief introduction to the different algorithms, encoding forms, and their classification performance.

2.2.1 Feature extraction

Substantial research in TLID feature extraction has been conducted in the decades since the 1960s. TLID features may be described as character- or word-level features. In the *Cambridge Dictionary*, the definition of a character is a 'mark or a symbol in writing, painting and other works'. Characters can be alphabetical, punctuation, numbers or other special symbols. Words are the minimal unit that is meaningfully written and pronounced in isolation. Both characters and words are sequential data

and easy to describe, so feature extraction from text is much easier than from audio and video. Besides these two basic feature extraction methods, some researchers have also tried to improve classification performance by using n-grams. Cavnar and Trenkle [1994] defined the n-gram as all possible co-occurring characters in a string within a particular language. Although n-grams can boost accuracy, they can be computationally expensive to collect, so the most commonly used n-gram models for TLID are unigrams, bigrams, trigrams and, if necessary, four-grams [Manning and Schütze, 1999].

Table 2.2: TLID feature extraction methods

Feature	**Citation**
Particular character detection	[Clive et al., 1994]
Particular word detection	[Ingle, 1976][Henrich, 1989] [Clive et al., 1994]
Particular character n-grams	[Henrich, 1989] [Clive et al., 1994] [Sibun and Reynar, 1996]
Frequency of character n-grams	[Beesley, 1988] [Henrich, 1989] [Cavnar and Trenkle, 1994]

Table 2.2 summarises the feature extraction techniques used in TLID. Current research tends to use character sequences, particular characters and word detection, especially in cross-language family identification. Most prefer to use Cavnar and Trenkle [1994]'s n-gram counting by using rank order statistical techniques; however, Dunning [1994] found that Bayesian models for character sequence prediction performed as well as Cavnar and Trenkle [1994]'s n-gram. Cavnar and Trenkle [1994] argued that n-gram models need tokenisation and this would be harmful to performance. He introduced Markov models to predict the probability distribution of characters because they are easy to deal with mathematically and can be described relatively succinctly. His work used Bayesian decision rules to minimise the probability of error of which possible language models (if there are more than two models) have caused a particular observing string. [Dunning, 1994] claimed that the accuracy of about 92% could be achieved with only 20 bytes of test text and $50K$ of training. This result could be improved up to 99.9% accuracy after testing 500 bytes.

[Ingle, 1976] emphasised the importance of language identification approaches. Previously, TLID was a task relatively unfamiliar to computer science but more closely related to linguistic problems. On receiving an unfamiliar language text, translators had to identify the language using a list with languages based on special features such as words and characters. Up to this point, TLID was seen as a specialist task that could only be tackled successfully by skilled translators. However, Ingle [1976] noted that it was possible to design a table that could implement language identification by unskilled operators using some 'key' words. By manually selecting the most frequently used words, he listed the single or two-lettered words of an unknown language text and eliminated languages that were the lowest probability for each word.

Although Ingle [1976] did not automate his method, Henrich [1989] realised his idea as a computer program in ASCII text. He believed that the identification system could use only information of character codes of written text, word length, and positions of words in specified sentences. By working on English, German and French through an n-gram model, the identification accuracy was 51.4% in unigram but much improved in trigrams to 73.6%.

Clive et al. [1994] tried to combine unique character detection, word frequency and n-gram models in his project. His results also showed that unigrams performed worse than bigrams, whose accuracy was 88%, which, in turn, was worse than trigrams, with an accuracy of 91%. He found that, for bigrams and trigrams, it was not necessary to learn all n-grams and only 75% of bigrams were required, and even fewer for trigrams, which only required $25 - 50\%$. Sibun and Reynar [1996]'s character unigram statistics or character bigram statistics results, based on the ISO Latin-1 alphabet, also support Clive et al. [1994]'s conclusion and Sibun and Reynar [1996] claimed that 100% accuracy was possible using n-gram models.

Beesley [1988], who found the relationship between character frequency and languages, argued that applying the highest n-gram likelihood to identify a language as possible. Cavnar and Trenkle [1994] continued his work by getting the highest

overall accuracy, which would be 99.8%, within 300 characters. Since then, the Cavnar and Trenkle algorithm, Cavnar and Trenkle [1994], has become the default algorithm for computer implementation. We will use Cavnar and Trenkle [1994]'s method for our text, audio and video language identification methods and show the identification results. Its technical details of n-gram frequency are presented in Section 3.2.1.1.

2.2.2 Algorithms

Table 2.3: Review of identification techniques

Techniques	**Citation**
Manual	[Ingle, 1976]
Support vector machines and kernel methods	[Henrich, 1989] [Clive et al., 1994]
Monte Carlo based sampling	[Poutsma, 2002]
Text compression	[Benedetto et al., 2002] [Cilibrasi and Vitányi, 2005]

Since Ingle [1976] provided a manual language identification table for his translation tasks, language identification had been developing into fully automatic analysis systems. Alternative algorithms, which were previously mentioned (Sibun and Reynar [1996]; Henrich [1989] in Section 2.2.1), are important recognition results. However, Henrich [1989] still needed to manually build exception character combinations in languages.

Poutsma [2002] described methods to address one of the disadvantages of language identification systems. He claimed that language identification systems needed too much data to train models and could catch only a few language features. He introduced a Monte Carlo sampling method to find which features appeared the most in languages. Hence, language identification tasks could reduce the number of samples required for training. He also used standard errors to check whether the database was large enough. He compared the performance of n-gram and common words techniques by applying Monte Carlo sampling and found that the n-gram's

results were much better than those of common words with less than 100 characters of input.

Benedetto et al. [2002] introduced an interesting technique by compressing text and measuring the Kolmogorov distance between pairs of texts in 10 official languages of the European Union. The Kolmogorov distance in this case, which is also known as Kolmogorov complexity, is the shortest length of computing program that produces as output the string. Based on this distance, he could build a phylogenetic-like tree to show the relationships between languages. The advantage of his technique is that the recogniser does not need to know the characters or n-gram information, which means it can work much more efficiently than other n-gram-model-based systems.The detailed technique is described in Section 3.3. Cilibrasi and Vitányi [2005] redid Benedetto's tasks and also achieved good performance.

2.2.3 Language selection

Sibun and Reynar [1996] noted that almost all recognition systems were built on ten or fewer languages due to lack of resources. Hughes et al. [2006] argued that one significant issue existing in text language identification studies was the paucity of data. Before Hughes et al. [2006], previous studies collected data from large numbers of sources for specific projects that were often too specific to be applied to other problems. The result was that research for a specified language might not have been suitable for other languages.

In addition, Hughes et al. [2006] noted the problems of 'open class language identification', which is the question of whether classifiers can guess unknown languages.

2.2.4 Text encoding and input format

Since the invention of Morse code, there had been a question of how to best encode text for machine translation.

The most common character coding used in modern techniques is the Unicode

character set [Chopra et al., 2005], yet most language identification systems have been designed to use the ASCII encoding [Sibun and Reynar, 1996].

Because ASCII (also Extended ASCII) coding is shorter than Unicode, it can only accommodate western European language coding, which limits the language selection. To fit the increasing requirement for a uniform code set, The Unicode Consortium [2011] reports the invention of a new character coding set like ASCII, but which could work for the whole world, called 'Unicode'.

Since the rapid expansion of the multilingual internet, Unicode and 8-bit Unicode Transformation Format (UTF-8) conventions later became the most popular website characters.

Later on, in proposal $98-18$: Unicode Identification and Encoding in USMARC records, [Aliprand, 2011] suggested the use of UTF-8, which is recognised as the best current practice by the Internet Architecture Board. Unlike previous character encodings that had strong relationships with a particular language, a Unicode character is a 16-bit entity and thus able to display over $65,000$ characters. It is an international charset that contains the most commonly used languages in the world. Table 2.4 displays the languages that Unicode currently supports.

Table 2.4: Unicode 6.2 Character Code Charts [Unicode, 2013].

European Scripts	Middle Eastern Scripts	South Asian Scripts
Armenian	Arabic	Bengali and Assamese
Coptic	Aramaic, Imperial	Brahmi
Cypriot Syllabary	Avestan	Chakma
Cyrillic	Carian	Devanagari
Georgian	Cuneiform	Gujarati
Glagolitic	Hebrew	Gurmukhi
Gothic	Lycian	Kaithi
Greek	Lydian	Kannada
Latin	Mandaic	Kharoshthi

Linear B	Old South Arabian	Lepcha
Ogham	Pahlavi, Inscriptional	Limbu
Old Italic	Parthian, Inscriptional	Malayalam
Phaistos Disc	Phoenician	Meetei Mayek
Runic	Samaritan	Ol Chiki
Shavian	Syriac	Oriya
Phonetic Symbols	Central Asian Scripts	Saurashtra
IPA Extensions	Mongolian	Sharada
Phonetic Extensions	Old Turkic	Sinhala
Modifier Tone Letters	Phags-Pa	Sora Sompeng
Spacing Modifier Letters	Southeast Asian Scripts	Syloti Nagri
Superscripts and Subscripts	Balinese	Takri
Combining Diacritics	Batak	Tamil
Combining Diacritical Marks	Cham	Telugu
Combining Half Marks	Javanese	Thaana
American Scripts	Kayah Li	Vedic Extensions
Cherokee	Khmer	Philippine Scripts
Deseret	Lao	Buhid
Unified Canadian Aboriginal Syllabics	Myanmar	Hanunoo
African Scripts	New Tai Lue	Tagalog
Bamum	Rejang	Tagbanwa
Egyptian Hieroglyphs	Sundanese	East Asian Scripts
Ethiopic	Tai Le	Bopomofo
Meroitic	Tai Tham	CJK Unified Ideographs (Han)

N'Ko	Tai Viet	CJK Compatibility Ideographs
Osmanya	Thai	CJK Radicals / KangXi Radicals
Tifinagh	Other	Hangul Jamo
Vai	Alphabetic Presentation Forms	Hangul Syllables
	Halfwidth and Fullwidth Forms	Hiragana
	ASCII Characters	Katakana
		Kanbun
		Lisu
		Yi

When the Unicode project began, the ISO 10646 standard was also simultaneously started by the Joint Technical Committee 1 (JTC1) of the International Organisation for Standardization (ISO) and the International Electrotechnical Commission (IEC). Unlike the ISO 10646 standard, Unicode defines character properties (script direction, punctuation, shaping, width, etc) and implementation rules. Both the ISO and Unicode character repertoire and encoding are successfully and accepted by HTML 4.0 Raggett et al. [1999], XML 1.0 Bray et al. [2008] and their later versions, and they are also able to map to previous encodings, including ASCII, easily.

As Unicode had 16bits code length, the Unicode Consortium realised they required double the space on disk rather than those coding sets that only need 8 bits. Unicode solves this problem through its flexible variable length coding.

The method of flexible variable length coding of UTF-8, UTF-16, and UTF-32 is a compressed stream of bytes, which means that space requirements will vary alongside based on encoding form. The Unicode compression schemes vary according to context with 10 encoding modes: single-byte mode, Unicode model, window,

locking shift, non-locking shift, dynamically positioned window, static window, tag byte, index byte and supplementary codespace [Wolf et al., 2000].

2.2.5 Length of text and evaluation methods

The final issue mentioned by Sibun and Reynar [1996] is a common one in machine learning - it is difficult to compare classifiers trained in different ways. Sibun and Reynar [1996] proposed that there are two main factors could be taken into consideration: length of training data and methodology complexity.

They reviewed a number of algorithms and found that, in almost all cases, the longer the training data, the better the classifier. Furthermore, some methods were very slow - is it reasonable to compare a slow, exhaustive method with a fast, effective one?

[Hughes et al., 2006] suggested building a standard evaluation corpus so that a variety of systems could be tested and compared to each other. She also noted the importance of the effects of pre-processing, which were ignored or not mentioned in most language identification research, although stemming, stop word removal, case folding, and other kinds of normalisation usually improve the results.

2.3 ALID(Audio Language IDentification)

Audio language identification (ALID) is a computer system that enables the recognition of a language based on its digitised speech signal [Zissman and Berkling, 2001]. As the world becomes more globalised, there is an increasing need to identify and translate the spoken language but it is unlikely that this need can be met with human exerts alone. ALID development could provide a faster and cheaper language classification system. Usually, the ALID application automatically detects the speaker's language and switches to the right language system or links to human interpreters. ALID would work well, particularly, when there is a large number of

speakers and it is difficult to find enough experts for artificial language identification.

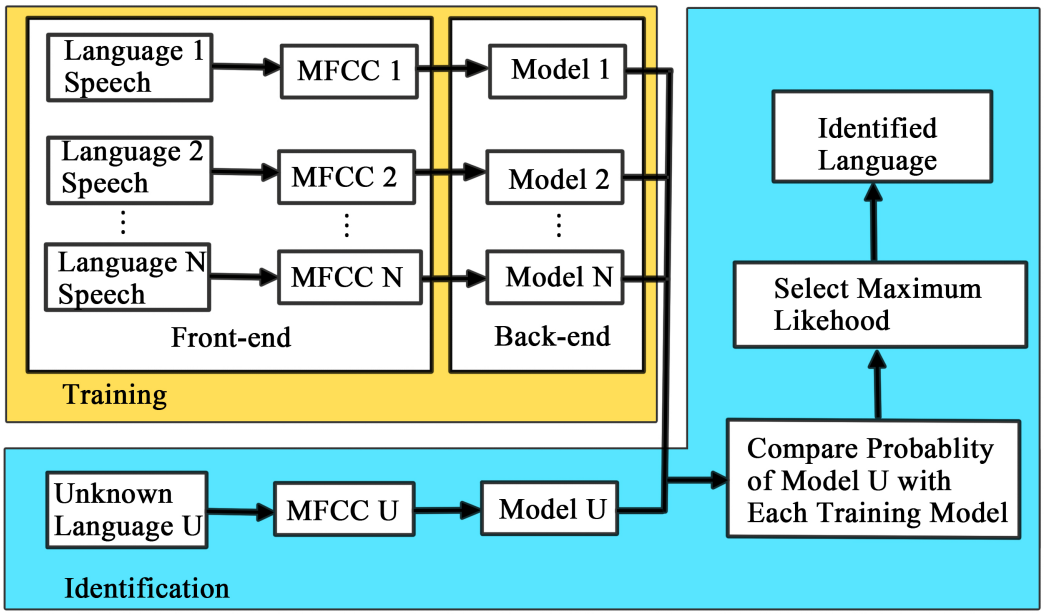


Figure 2.1: Basic language identification system procedure using MFCC features [Ambikairajah et al., 2011]

Figure 2.1 shows a basic ALID system using MFCC features. Generally, an ALID system can be separated into two processes: training and identification.

For the training stage, the system can be further divided into the front-end and back-end sections. The front-end section works mainly on extracting features from speech data; the back-end section builds models for language feature vectors.

Speech data in ALID refers usually to acoustic information. Because speech data is too large to be analysed directly, features are used to summarise the data, so feature extraction is essential for feature discrimination. Based on the principles of phonetics, the features of languages can be utterance level, syntax level, morphology level, and others. Typical features in most ALID projects include Mel-frequency cepstral coefficients (MFCCs), linear prediction coefficients (LPC), perceptual linear prediction (PLP), learner prediction cepstral coefficient (LPCC), smart data compression (SDC), etc. [Ambikairajah et al., 2011] (The definition of these features will be explained in Section 2.3.1). However, all of the methods focus on extracting speech features as much as possible while losing minimal information. Section 2.3.2

discusses feature extraction in detail.

Having extracted feature vectors from the speech data, the back-end section builds a model to describe the features for each language. During the identification stage, the ALID system extracts an unknown language feature from the speech data and builds up a language model for it. The ALID system then uses some measurements to compare similarities between known and unknown language models and identifies the languages based on maximum likelihood probability. For example, if an unknown language is classified as 70% English and 30% Italian, the system would recognise the unknown language as English.

2.3.1 Speech information

Based on the work of Ambikairajah et al. [2011], speech features for language identification could be roughly categorised as low and high level. Low-level features contain acoustic, phonotactic and prosodic information. High-level features include morphology, syntax and grammar information.

Although acoustic, phonotactic and prosodic features are all low level and could be extracted directly from speech data, there are different levels of analysis of speech production. The acoustic level is the initial production of the analysis of speech data and is closest to the physics of the real speech data. To judge whether two acoustic level events are different, the instrumental acoustic analyser should be able to provide evidence of the differences. Considering timing and quality, either two repetitions made by a single speaker but linguistically and paralinguistically identical, or two utterances made by two different speakers would be thought of as different at the acoustic level[Laver, 1994]. The timing and quality differences between the utterances in a digital signal could be represented as differences along an ordinate or amplitude of the waveform. In practice, phonotactic- and prosodic-level features could be extracted from acoustic-level speech data through MFCC, LP, PLP, LPCC, acceleration cepstrum and SDC [Ambikairajah et al., 2011].

As a branch of phonology, the phonotactic level concerns the constraints in syllable structure and phonological distribution of consonant and vowel. Different languages would have unique consonants and vowels and an individual consonant or an individual vowel could occur in different positions phonologically. The example given by Laver [1994] is the word 'zloty', which cannot be recognised as English because the position of the consonants /zl/ never appears as a bigram.

The prosodic level is related to articulation, phonation and overall muscular tension factors [Laver, 1994]. The main prosodic areas that are studied by ALID are tone, stress, duration and rhythm [Ambikairajah et al., 2011].

As we described in Section 2.2, lexical morphology and syntactic structure are high-level features concentrating on the language structure itself. Based on investigating the internal structures of words, lexical morphology also includes describing how many similarities there are between words such as *happy* and *happiness*. It is not difficult to recognise words in different languages because the word components, such as the root, the prefix and the suffix (collectively known as affixes), are always different in each language. Additionally, different languages also have their unique word dictionaries and ways to form words.

The syntactic level is more concerned with the words used in languages. For example, most spoken languages have their unique word vocabularies that could be used in ALID. In ALID, the most used word-level features are morphology and syntax, which have been discussed in Section 2.2.

In conclusion, low-level features are easier to obtain compared to high-level features, while high-level features might be able to extract more language-discriminative information. However, high-level feature extraction spends more time analysing large datasets and finding lexical and syntactic rules between phrase, clauses, and sentences.

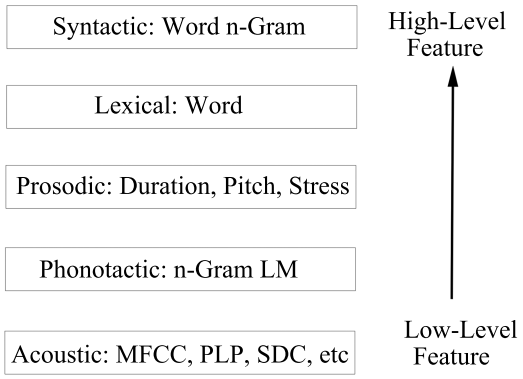


Figure 2.2: Basic language identification system procedure [Ambikairajah et al., 2011].

2.3.2 Feature extraction

In Figure 2.2, acoustic features are the lowest and most simple level. Speech events at this level can be distinguished through the amplitude and frequency components of waveforms. Acoustic features normally include Mel frequency cepstral coefficient (MFCCs), linear prediction coefficients (LPCs), perceptual linear prediction (PLP), and linear prediction cepstral coefficient (LPCCs). Sometimes, these features are augmented with additional information such as delta and acceleration cepstrum and smart data compression (SDC) [Ambikairajah et al., 2011].

The phonotactic feature sets rules for the sequence of admissible sound patterns. Obviously, not all phonemes appear in all languages, which means it is possible to identify a language by its phonotactic features. For example, the sequence \zl-\does not appear in native English pronunciation so *zloty* cannot be a native English word. Thus, a speech containing *zloty* could never be identified as English. Tong et al. [2006] support the use of phonotactic features since they might have a better performance than acoustic features and are less complex than other high-level features. One phonotactic feature analyser suggested by Ambikairajah et al. [2011] is the N-gram language model. The technical details of the N-gram model will be discussed in Section 3.2.1.1.

Prosodic features are not encoded by grammar, and generally have not been handled fully in LID [Tong et al., 2006]. Current ALID prosodic features include duration, pitch, rhythm and stress of language. To present features in quantitative digital signal format, the tone is explained as the pitch or fundamental frequency, stress is defined as intensity and rhythm as a duration sequence. Prosodic features would be of help in identifying tonal languages such as Mandarin, Thai or Vietnamese, or language with stresses such as English, French and Spanish. Moreover, prosodic features are identical in emotional information such as rising tones [Ambikairajah et al., 2011].

Syntax features specify the rules of forming phrases, clauses and sentences. Grammar is one of the typical sentence-generation rules. Note that the same word might exist in different languages but once put into context, it would be much easier to determine to which language it belongs [Zissman, 1996].

2.3.3 Recognition approaches

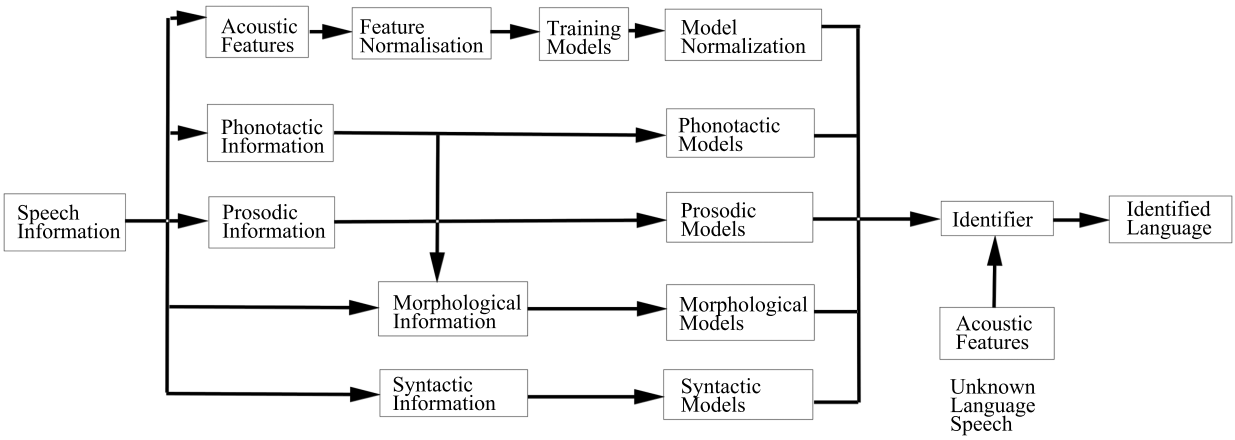


Figure 2.3: ALID system processing detail [Ambikairajah et al., 2011]

.

An ALID system normally consists of two components: training processing and identification processing. By training on acoustic-level features, an ALID front-end system builds one or more compact and representative models, which describe

language-dependent, fundamental speech properties [Zissman, 1996].

In Figure 2.3, the front-end system contains four steps: data pre-processing, feature extraction, appended features and feature normalisation [Ambikairajah et al., 2011].

The pre-processing steps include voice activity detection, windowing and pre-emphasis. Voice activity detection detects signal existence and eliminates extraneous information from the signal. Windowing, sometimes called apodisation and tapering, sets a zero value on a junk interval. Pre-emphasis techniques refer to frequency selective amplification to improve part of the magnitude in order to enlarge the overall signal-to-noise ratio. Pre-processing can also analyse the data to check that it is useful for training - a phase known as data validation.

The feature extraction step parameterises the signal into numerical vectors and provides much of the most useful data for distinguishing languages.

Appended features, commonly used, such as delta features and smart data compression (SDC), are added into the feature vectors. A previous study by Bielefeld [1994] suggests that appending SDC features could improve ALID system performance.

The final step, feature normalisation, involves adjusting the trained models and improving the robustness by reducing noise and channel mismatch.

Zissman [1996] compared four commonly used approaches for ALID speech utterance systems: Gaussian mixture model (GMM) classification; single-language phone recognition followed by language-dependent and interpolated n-gram language modelling (PRLM); parallel PRLM (PPRLM), and language-dependent parallel phone recognition (PPR).

The GMM method is based on the assumption that different languages have different sounds and frequencies [Zissman, 1996]. The GMM function (2.1) is a parametric representation of a probability density function, based on a weighted sum of Gaussian densities of specified mean and variance [Gold et al., 2011].

$$p(\vec{v}_t|\lambda) = \sum_{k=1}^{N} p_k b_k(\vec{v}_t; \lambda). \tag{2.1}$$

where λ is the set of model parameters

$$\lambda = \{\vec{\mu}, \Sigma_k\}. \tag{2.2}$$

k is the mixture index ($1 \leq k \leq N$), p_k are the mixture weights and $\sum_{k=1}^{N} p_k = 1$ and b_k are the Gaussian densities function defined by means $\vec{\mu}$ and variance Σ_k.

In Zissman [1996], the maximisation of likelihood to determine the optimal parameters was performed via multiple iterations of the estimate-maximise (E-M) algorithm, as in [Dempster et al., 1977] and [Baum, 1972].

GMMs are computationally efficient compared to the other three systems and do not require orthographically or phonetically transcribed speech data. However, GMMs perform worse than single-language phone recognition followed by language-dependent and interpolated n-gram language modelling (PRLM), PPRLM, and language-dependent PPR [Zissman, 1996].

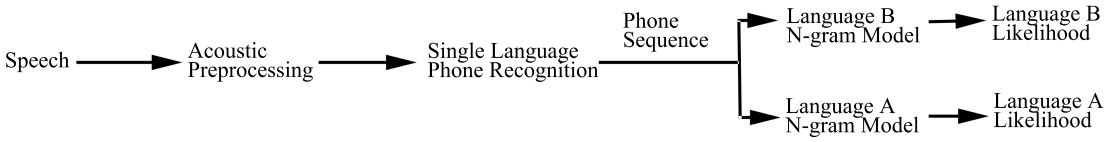


Figure 2.4: PRLM system processing detail [Zissman, 1996]

.

A single-language phone recognition followed by language-dependent and interpolated n-gram language modelling (PRLM) procedure is shown in Figure 2.4. It applies a single model combined with n-gram features that are labelled depending on a single-language recogniser output rather than human-supplied orthographic or phonetic features. In testing, speech is tokenised and, based on its symbol sequence

likelihood of each language, its highest likelihood is identified via the corresponding n-gram model.

Since PRLM is not a language-dependent system, in some cases, it can be trained on any language without a transcript or other high levels of information. The recogniser built by Zissman [1996] applied the hidden Markov model and the probability density modelled by a GMM counts the occurrence of n-gram symbols (usually, symbols mean phones). [Zissman, 1996]'s testing also found that there was only a small advantage to using $n > 2$ models.

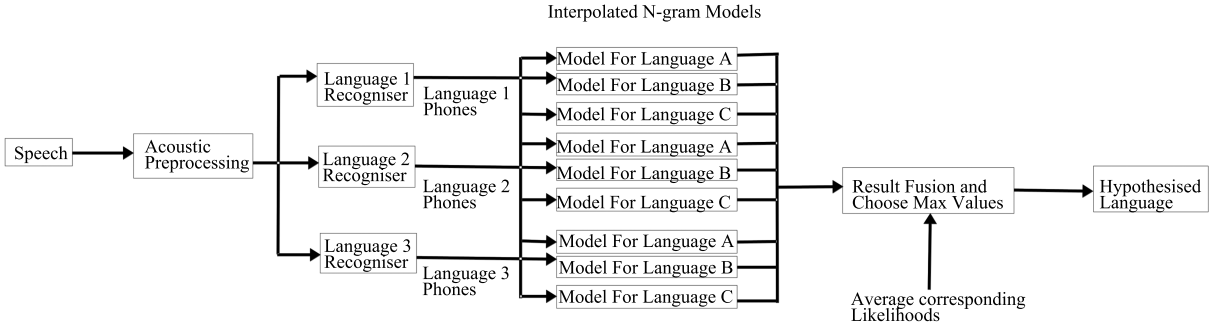


Figure 2.5: PPRLM system processing detail [Zissman, 1996]

.

Zissman [1996] defined parallel PRLM (PPRLM) as an improvement of PRLM. Since the sounds that are unique in a language might not always occur in speech, especially when a short time of identification is required, it is better to train multiple language recognition models in the same system.

Figure 2.5 shows an example of a PPRLM system. First, unique models for target languages are built in each language front-end system. Then, the PPRLM system calculates the highest likelihood overall and obtains the hypothesised language. Although both PRLM and PPRLM system performance is better than a GMM system, the number of training language limitations remains a concern. The more languages trained, the more time required for training models and identification.

PPR runs a single-language phone recognition in parallel, and, like PRLM but

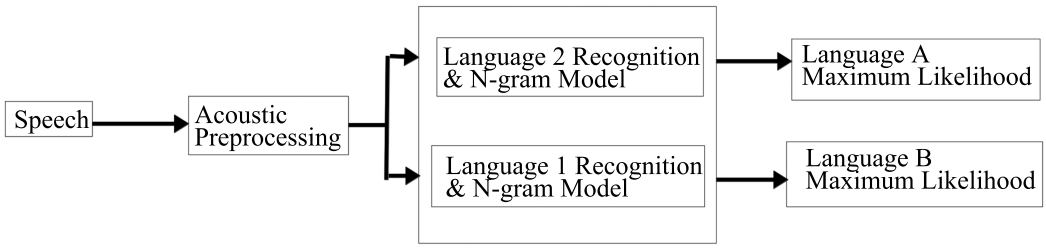


Figure 2.6: PPR system processing detail [Zissman, 1996]

.

instead of HMM, its inter-phone transition between phone i and j is

$$a_{ij} = s \log \widetilde{P}(j|i) \tag{2.3}$$

where s is the grammar scale factor, and P are the bigram probabilities derived from training speech labels. He also points out that PPR differs from PRLM, as PPR recognises phones by using the 100 most commonly used right context-dependent phones in addition to monophonic.

2.3.4 Normalisation

It is certain that noise will affect language identification performance. Normally, noise refers to environmental and mechanical noise. Environmental noise might be caused by other talkers, music or other factors, and mechanism noise might arise because of microphone quality and bad noise reduction techniques. In addition, the different volume of noise across records also has an impact on recognition results. Other factors that could contribute to unreliable recognition results, such as short-term channel distortions, speaker variations and other forms of interference, are proposed by De La Torre et al. [2002].

Normally, language recognition systems require data pre-processing before training and testing models. Normalisation techniques are necessary for any front-end language recognition system. The reduction of noise effects without the loss of in-

formation on acoustic characteristics is a desirable outcome for language recognition systems [De La Torre et al., 2002].

Some of the common normalisations are shown in Table 2.5.

Table 2.5: Review of normalisation techniques [Ambikairajah et al., 2011] .

Techniques	**Citation**
CMN(Cepstral Mean Normalisation)	[Atal, 1976]
Feature Warping	[Pelecanos and Sridharan, 2001]

De La Torre et al. [2002] assert that the main impact of noise is that it shifts the mean of the probability distributions of the features.

Cepstral mean subtraction (CMN) was introduced by [Atal, 1976]. Cepstral mean and variance (CMVN) normalisation for mismatching means of the probability distributions of the features, has become a standard method and is widely applied in ALID systems for its simplicity.

Current ALID systems use Mel frequency cepstral coefficients (MFCCs) as features. MFCCs are derived from speech waveforms by a log filterbank, which means that non-linear transformation caused by additive noise also exists in MFCCs [De La Torre et al., 2002]. Since CMN is not good at dealing with the non-linear distortion effects on the cepstral coefficients, [De La Torre et al., 2002] suggests adding the histogram equalization (HEQ) technique to improve the transformation quality.

Feature warping is a linear normalisation method. To deal with additive noise and mismatched channel environments, Pelecanos and Sridharan [2001] recommend filtering noise through a channel, which known as feature warping. It constructs a more robust speech feature distribution by transforming individual cepstral feature streams.

2.3.5 Current problems

There are a number of identified problems with multi-lingual speech identification.

1. The limited amount of multi-lingual speech data available for training the

automatic LID system. Researchers may not want to share their data for security or financial reasons, among others. Systems trained and tested on different datasets are not comparable. In addition, other researchers collect their own data, then this is a wasted effort to compare with other researches by using different datasets.

2. The limited number of languages that the current systems are able to identify (typically 10-15 out of about 6,900 'living' languages). Unless there are large amounts of speakers available, it is difficult and expensive to collect small languages and build a large data corpus for them. Therefore, most current studies focus only on certain high-frequency languages such as English, Arabic, French, Russian, Mandarin, etc. See Table 2.1.

3. The currently limited incorporation of different dialects within the same language also make speech recognition difficult to realise. Although we use normalisation to minimise the differences between independent speakers, we cannot remove the dialect impact from the speech data.

4. Another significant deficiency in most current systems is that they perform well on 30- or 45-second samples but relatively poorly on shorter $3-10$-second samples. In an emergency situation (such as a call to an emergency number), a shorter identification time is essential.

2.4 VLID (Visual language IDentification)

Compared to TLID and ALID, VLID is new and so has received less attention. Sumby and Pollack [1954] concluded that visual information contributed to speech intelligibility. Due to technology limitations, their work was mainly focused on humans rather than machines. However, their conclusions indicated that visual information was promising at low speech-to-noise ratios and the relative visual information extracted from speakers' facial and lip movements was not related to speech-to-noise

ratio. Sumby suggested that visual information would work in many practical situations such as those encountered in military or industrial applications. Starner [1995] mentioned VLID as a good way to identify sign language that does not have any speech information. It might also be helpful in situations such as noisy environments when the talker is distant from the microphone.

2.4.1 VLID models

Petajan built the first audio-visual speech recognition in 1984 [Matthews, 1998]. This system illustrated that visual recognition improved the recognition accuracy on digits, letters and some small vocabulary tests. Later on, Petajan's improved system introduced the vector quantisation of the mouth images and dynamic time warping to better align with utterances for template matching.

2.4.2 Visual feature extraction models

Table 2.6: Historic review of feature extraction techniques

Year	**Technique**	**Citation**
1974	DCT(Discrete Cosine Transform)	[Ahmed et al., 1974]
1984	Eigenlips	[Bregler and Konig, 2002]
1987	Snake	[Kass et al., 1988]
1995	ASM(Active Shape Model)	[Cootes et al., 1994]
1998	AAM(Active Appearance Model)	[Cootes et al., 2001a]
1998	MSA(Multiscale Spatial Analysis)	[Matthews et al., 1998]

The original lip-reading systems were based on a low-level analysis of images such as the discrete cosine transform (DCT) but later works attempted to capture shape explicitly. Kass et al. [1988] first attempted 'snakes' to fit an elastic contour to greyscale contours. Unfortunately, the greyscale gradient from the lips and face display results proved to be too weak to train a snake [Cetingul et al., 2006].

The AAM aims to solve the boundary definition problem. So far, as Matthews proposed in 1998, attached AAM is generally regarded as the most effective method for tracking and feature extraction in the lip area[Newman, 2011].

Table 2.6 shows the history of techniques used for feature extraction. There are six types of model: discrete cosine transform (DCT), Eigenlips, AAM, ASM and multiscale spatial analysis (MSA), which are in common use today.

The DCT can provide very efficient compressive information in the fewest coefficients. However, the disadvantage of the 2D DCT is that it is very sensitive to changes in illumination when used for feature extraction [Aguilar-Torres et al., 2009].

Eigenlips was proposed by Turk and Pentland [1991] for facial recognition purposes. It is a machine learning method and is able to work under variable conditions since it sets up parameters to define the spatial shifting, rotation and scaling, and also captures the lighting.

MSA is a low-level, pixel-based method that does not relate to the absolute amplitude or the positions of images, so, again, it is a very fast and robust method of feature extraction [Matthews et al., 2002, 1998].

ASM is another high-level, model-based method for lip-reading feature extraction from image sequences. Matthews et al. [1998] argued that it compactly describes the shape of the lips with several contours, and is able to display the lips' movement in detail.

AAM is an improvement on ASM since the shape-only analysis is insufficient for facial recognition. Both AAM and ASM use a (top-down) model but AAM also uses appearance alongside shape models [Matthews et al., 2002]. The other significant advantage of AAM is that it can also fit the emotions the speakers show [Cootes et al., 2001a].

Although AAM and ASM are popular in current studies, the concern over computing efficiency still exists. Both precompute several images as samples and use them to work out the result by updating an iterative matching algorithm. This procedure, which transfers from high-dimension pixels to low-dimension computational metrics, can be moderately expensive.

2.5 Current problem

Both language typologies are widely accepted; however, this makes it difficult for computational language classification to find a background truth for comparison. Although the language tree gives the differences between languages, it does not tell us the distances between two languages. For example, in Figure 2.7, is Catalan further from Spanish than Portuguese? We also cannot tell how far English is from Chinese because there is no distance definition between Indo-Hittite and Sino-Tibetan. What is more, the linguistic language is more text based than audio and video, so is the linguistic language tree reliable for audio and video language identification?

One of our tasks is to try to build up language trees based on text, audio and video and compare them with the linguistic language tree. Since we have chosen the linguistic language tree as the background truth, we will find the differences between the trees using the Robinson-Foulds tree distance measurement.

2.5.1 Linguistic language tree classification

For human languages, linguistics concludes that there are two kinds of language classification that are most helpful for researchers. One is the classification of languages by typology and the other is what has been called genetic classification.

Typology classification concentrates on the structure of languages as phonological and grammatical complexity similarities. Examples of these similarities might be the number and kinds of vowels and the order of sentences. Although the typology classification could explain the relationship between languages, it is not a perfect method because the grammar derivations between languages are not fully understood. There is no guarantee that the inadequate information in phonology and grammar would not impact on the classification [Voegelin and Voegelin, 1977].

Genetic classification divides languages by generations. Like typology classification, genetic classification takes the grammatical and phonological similarities into

consideration, not only introducing the original languages from which other languages are descended but also considering the historical background relationships. Genetic classification is widely accepted by linguists. The concept of the 'language tree' was introduced into classification to represent the generations and closeness between languages. Figure 2.7 describes the genetic classification of the main Indo-Hittite languages. Each language has its own historical origins and links to other languages by their common origins. Figure 2.8 shows other language trees for languages we are going to use in our project [Ruhlen, 1991].

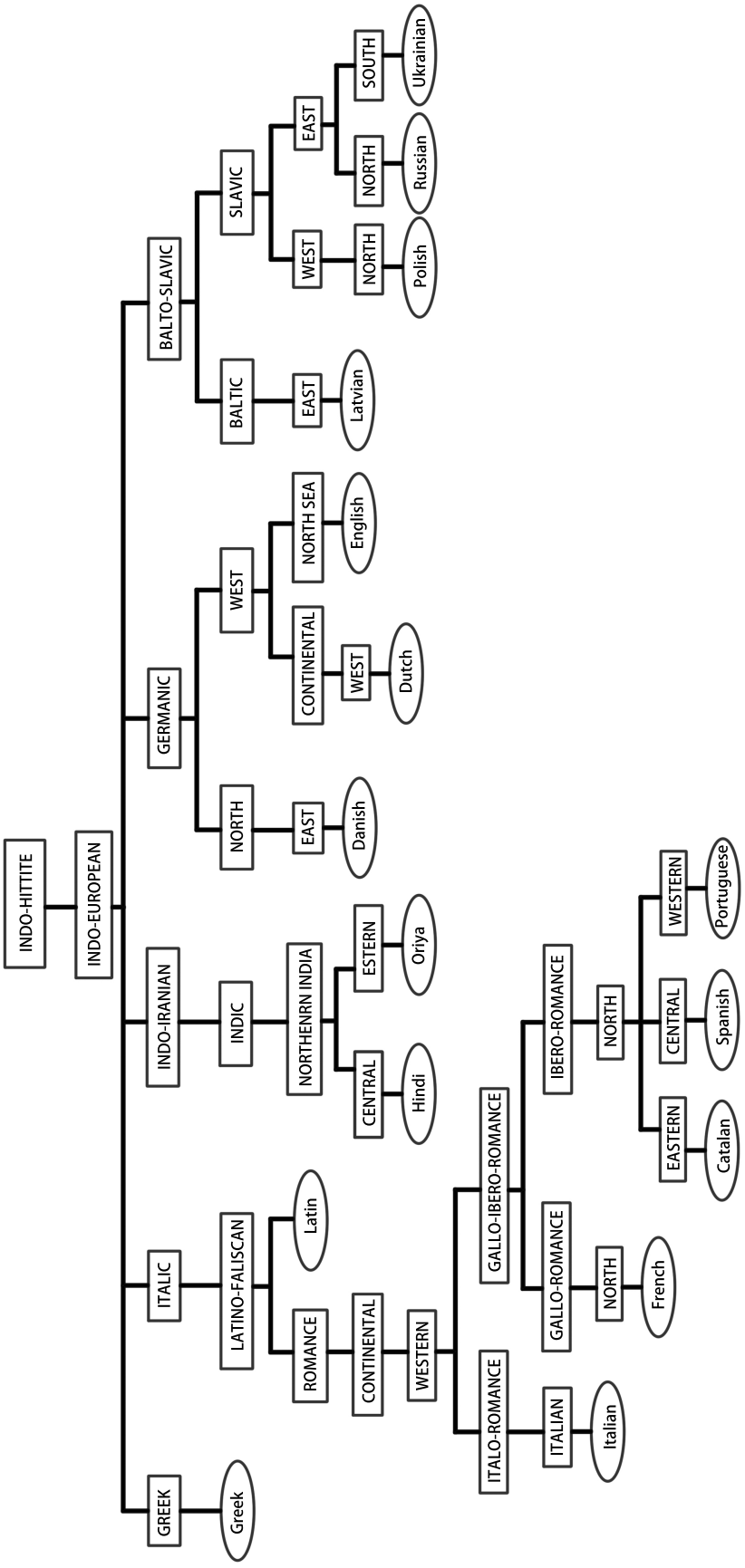


Figure 2.7: Language tree of European family

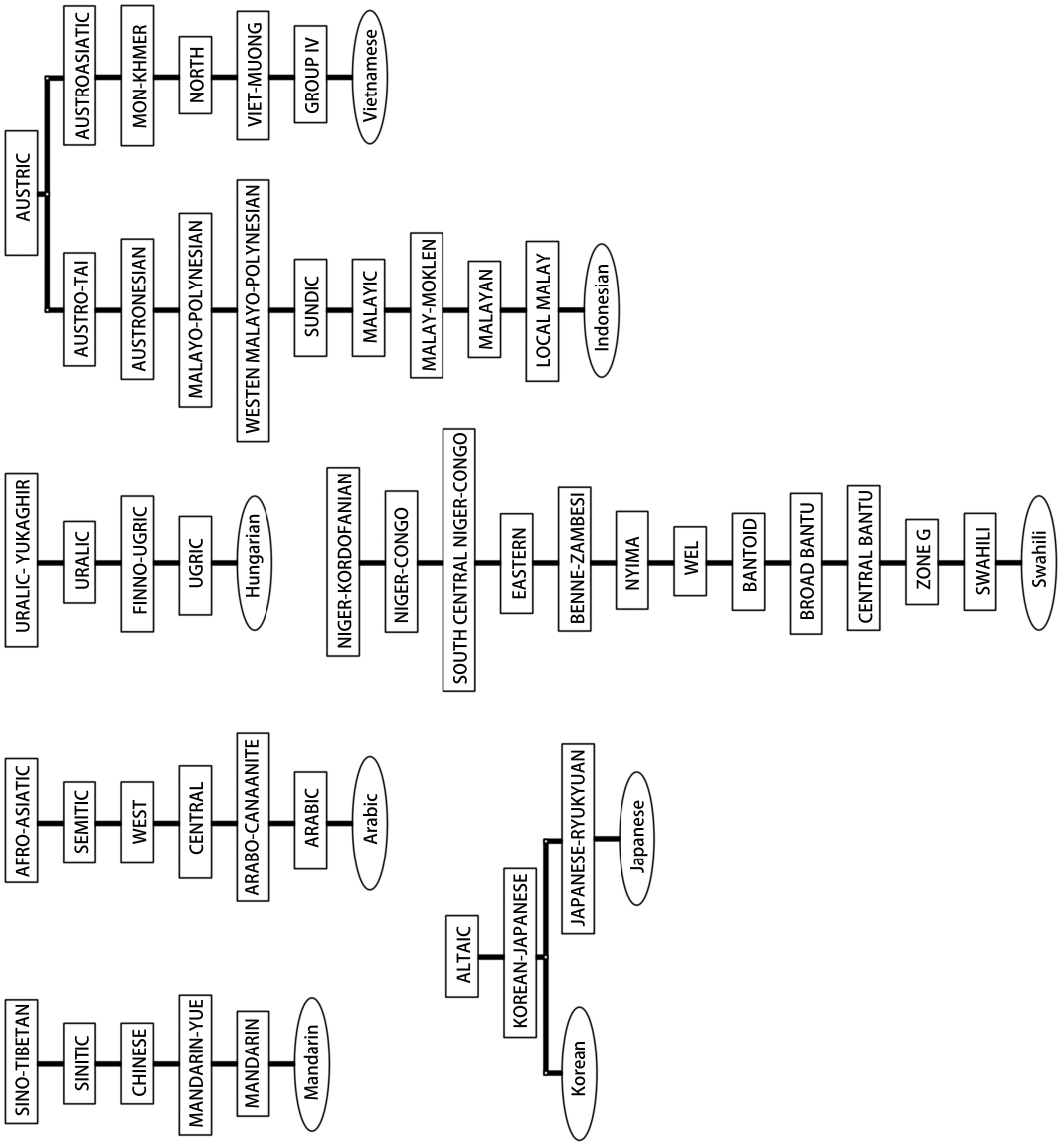


Figure 2.8: Other language trees we are going to compare languages.

Ruhlen [1991] gives two definitions of genetic language classification. One is that all relevant languages should be categorised into one subclass and the other is that the languages in one subclass should be related more to each other than to languages outside the class. In Figure 2.7, it is clear that Portuguese, Spanish, Catalan, French and Italian are under the Latin class and are more related than other languages such as Japanese, Arabic, Chinese and so on.

2.5.2 IPA (International Phonetic Alphabet)

The IPA (International Phonetic Alphabet) aims to find a consistent method to represent the sounds of language and display them mainly by the Roman alphabet. It is widely used in dictionaries and acoustic annotation. The International Phonetic Association claims that the IPA is comprehensive enough to understand nearly all sounds in all languages in the world [Association et al., 1999]. IPA contains 107 letters, 52 diacritics and 4 prosodic marks. Table 2.7 shows the IPA chart which includes the basic IPA letters.

Table 2.7: International Phonetic Alphabet [International Phonetic Association, 2018].

Consonants (pulmonic)											
	Bilabial	Labiodental	Dental	Alveolar	Postalveolar	Retroflex	Palatal	Velar	Uvular	Pharyngeal	Glottal
Plosive	p b		t d			ʈ ɖ	c ɟ	k g	q ɢ		ʔ
Nasal	m	ɱ	n			ɳ	ɲ	ŋ	N		
Trill	B		r						R		
Tap or Flap			ɾ			ɽ					
Fricative	ɸ β	f v	ð θ	s z	ʃ ʒ	ʂ ʐ	ç ʝ	x ɣ	χ ʁ	ħ ʕ	h ɦ
Lateral Fricative			ɬ ɮ								
Approximant		ʋ	r			ɻ	j	ɰ			
Lateral approximant			l			ɭ	ʎ	ʟ			
Consonants (non-pulmonic)			Vowels)			Other symbols)					
ʘ	b	’	i y	ɨ ʉ	ɯ u	ʍ	ɥ	ʜ			
ǀ	d	pʼ	e ø	ɘ ɵ	ɤ o	ɧ	ʡ	ʢ			
ǃ	j	tʼ	ɛ œ	ɜ ɞ	ʌ ɔ	ʑ	ɺ	ɧ			
ǂ	g	kʼ	a ɶ		ɑ ɒ	k͡p					
ǁ	G	sʼ									

2.6 Current datasets

2.6.1 Text datasets

Our text data is based on the Universal Declaration of Human Rights (UNDHR). The UNDHR is provided by the United Nations General Assembly and freely available on the official website. It contains about 500 translations that are interpreted[1].

We use 255 text language scripts into our $n-$gram experiment. The list of text language datasets is in Appendix A. Table A.1 shows a list of all used languages with their codes for the representation of names of languages by ISO 639-2, ISO 639-3 and ISO 639-6 codes.

2.6.2 Audio datasets

For audio, we use the audio version of the UNDHR corpus. It is an open online resource provided by LibriVox and all records are uploaded by volunteers. Each record lasts more than ten minutes and is read by one female speaker. The audio is recorded using a 16-bit signal sampled at 22.050 kHz. Table 2.8 shows the list of languages that the LibriVox dataset provides[2].

Table 2.8: LibriVox datasets.

Arabic	Portuguese	Cantonese	Czech	English	Farsi
German	Hindi	Hungarian	Indonesian	Italian	Japanese
Korean	Mandarin	Polish	Russian	Spanish	Swahili
Swedish	Tamil	Vietnamese			

According to Figure 2.7 and Figure 2.8, we can find the Portuguese and Spanish are under the same sub-tree, Czech and Polish are under the same sub-tree and Japanese and Korean are under the same sub-tree. It means these three pairs of languages are linguistically closed to each other and can be used as criteria to evaluate our language distances results.

[1]http://www.ohchr.org/EN/UDHR/Pages/SearchByLang.aspx.
[2]https://librivox.org/the-universal-declaration-of-human-rights-by-the-united-nations/

The other dataset we use is Dr. Jacob Newman's dataset which is created in 2011 and contains three languages: English, Mandarin and Arabic. Unlike the LibirVox only has one speaker for each language, Jake collects multi-speaker for each language. The English dataset contains 24 speakers with 20 male speaker and 4 female speakers. The Arabic dataset has 9 speakers with 7 male speakers and 2 female speakers. The Mandarin dataset has 22 speakers with 9 male speakers and 13 female speakers. The audio files are recorded by 16-bit signal sampled at 22.050 kHz.

2.6.3 Video datasets

For the video datasets, we still want to compare to text and audio datasets. The video version of the UNDHR dataset was collected by Dr. Jacob Newman in 2011. It contains three languages: English, Arabic and Mandarin. These three languages are very different from each other. The speakers are recorded via high-definition video of the mouth region. Figures 2.9, 2.10 and 2.11 show some examples of the video images. Figure 2.9 shows Arabic speakers, 2.10 are Arabic speakers and 2.11 are Mandarin speakers. They were all recorded with 1920×1080 screen resolution, 48 KHz, stereo and 60 frames per second. The features of the videos are extracted by AAMs.

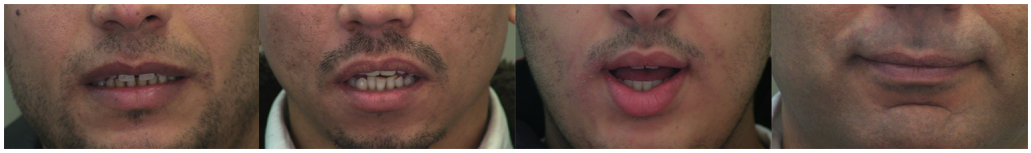


Figure 2.9: Arabic speaker examples that were recorded by video and only taken for mouth area movement.

Figure 2.10: English speaker examples that were recorded by video and only taken for mouth area movement.

Figure 2.11: Mandarin speaker examples that were recorded by video and only taken for mouth area movement.

Chapter 3

TLID (Text Language IDentification) results

3.1 Introduction

TLID is a mature field of research and is in routine use in high-traffic applications such as Google Translate. We describe two TLID techniques to evaluate their identification performance. We first use Benedetto et al. [2002]'s zipping language distances and then we compare the zipping results with an improved n-gram model that is based on the frequency idea of Cavnar and Trenkle [1994].

In this chapter, we will discuss the TLID procedures for zipping and n-gram modelling. For both the training and testing stages, we used scripts from the United Nations' Declaration of Human Rights and all scripts were encoded as Unicode. For the zipping method, we used the 17 languages are shown in Table 3.1.

Table 3.1: Languages used for Benedetto et al. [2002]'s zipping methods.

Vietnamese	Arabic	Chinese	Czech	English	German
Hungarian	Indonesian	Italian	Japanese	Korean	Polish
Portuguese	Russian	Spanish	Swahili	Swedish	

For the n-gram frequency model of Cavnar and Trenkle [1994], we applied all text

languages described in Section A.1. One of the advantages of the n-gram frequency model is that it only compares the highest-frequency grams and can quickly generate a language distance matrix.

3.2 Cavnar and Trenkle's N-gram model

3.2.1 Methods

Notwithstanding the recent interest in classification via zipping, the fact remains that the current state-of-art is Cavnar and Trenkle [1994]'s method, as previously explained in Section 3.2.1.1. To compare two languages, we rank all possible n-grams by their frequency and compare the difference between the two n-gram lists. For example, 'a' is ranked first in language A but ranked 10-th in language B, then the n-gram difference would be 10 between language A and B. If an n-gram is not in one of the languages, we set up a maximum penalty, which is conventionally chosen as 400.

We also use 10-fold cross-validation in this task. For the 17 language scripts, each language script is split into 10 parts. 9 parts are used for training the n-gram model and a single-fold script is used for identification. We then calculate the identification accuracy for each language and this step is repeated 10 times.

To examine the order of n-grams suitable for TLID, we introduce unigrams, bigrams, trigrams, quadgrams and five-grams in our task. Figure 3.1 shows the steps for implementing Cavnar and Trenkle [1994]'s N-gram model.

The dataset we used in this section was the UNDHR text dataset, which contains 254 languages; these are all available on the website[1]. Table A.1 in Appendix A shows a list of all languages used, with their codes for the representation of names of languages according to ISO 639-2, ISO 639-3 and ISO 639-6 codes.

[1]http://www.ohchr.org/EN/UDHR/Pages/SearchByLang.aspx

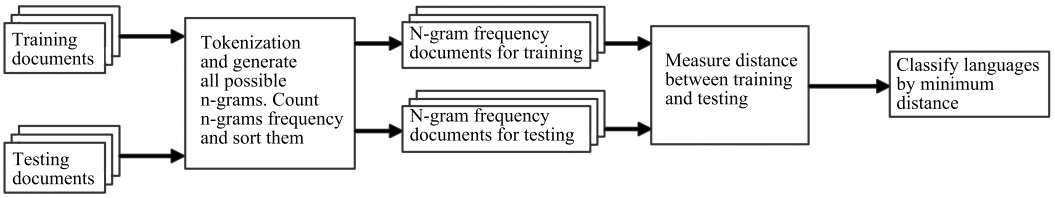


Figure 3.1: Cavnar and Trenkle [1994]'s n-gram frequency model for UNDHR dataset provided by librivox.

3.2.1.1 N-gram

In most automatic language processing applications, we will find the testing sentences have never been heard before. The n-gram model is a widely used method to solve this problem by predicting probabilities p [Manning and Schütze, 1999]:

$$p(w_n|w_1, ..., w_{n-1}) = p(w_1)p(w_2|w_1)...p(w_n|w_{n-1}), \tag{3.1}$$

which means the probability of gram w_n is based on the previous $n-1$ grams. So, after learning a lot of text, the model know which characters tend to follow other characters. Considering efficiency and accuracy, the n-gram model usually limits $n = 1, 2, 3, 4$, (or the unigram, bigram, trigram and four-gram model [Manning and Schütze, 1999]).

For the n-gram model, it is not necessary to divide sentences into words. Other features such as characters and phonemes could also be used in TLID.

The N-gram model used by Cavnar and Trenkle [1994] represented Zipf's Law, implying that a language could be identified by a set of high frequency words. The zipf's Law says that if the n-grams are list in order by the frequency of occurrence, for each n-gram, the frequency of occurrence is proportional to its position in the list [Zipf, 1949]. Cavnar and Trenkle [1994] used multi-length n-gram models simultaneously and also included blanks to the beginning and ending of the strings. For example, the word "GRAM" would be constructed by n-grams shown in table 3.2:

Table 3.2: n-gram construction for word "GRAM".

uni-gram:	_, G, R, A, M
bi-grams:	_G, GR, RA, AM, M_
tri-grams:	_GR, GRA, RAM, AM_, M__
quad-grams:	_GRA, GRAM, RAM_, AM__, M___

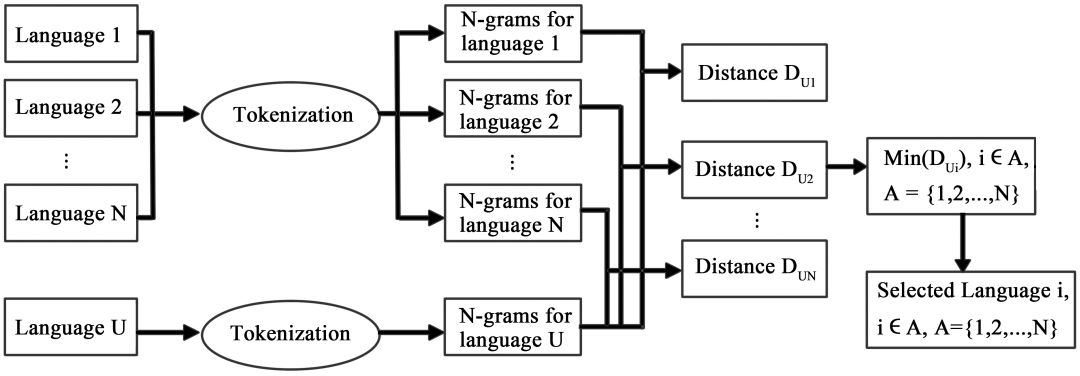


Figure 3.2: N-gram based language identification

Figure 3.2 shows the process of n-gram based language identification. Cavnar and Trenkle's n-gram model system first introduces a set of language files. Those files are then tokenized into single words, characters or phonemes from which we could construct n-grams. After generating n-grams and sorting them from high frequency to low frequency for each sample language, the system repeats the same steps for the unknown language U. The unknown language is classified as language i such that $i = avgmin(dist)$ when $dist$ is the distance between the n-gram of language U and the n-gram of languages.

The distance measurement used in Cavnar and Trenkle [1994] is the "out-of-place" rank-order statistic. For example, the bigram "pre" is at rank 5 in the sample language k while it is at rank 3 in the unknown language U, then the "out-of-place" value between language k and U is 2. If there are some n-grams found in neither of the languages, then the system uses a predefined maximum "out-of-place" value. The distance between two n-gram sequences is then the sum of the "out-of-place" scores with all n-grams. We used "penalty" for short in this thesis.

Since Cavnar and Trenkle's n-gram model sorts n-grams from high frequency to

low frequency, it does not need to worry about the specific frequency thresholds or the distributions at a specific range of values. Such an n-gram system has proven to provide a high accuracy language identification solution by simply collecting a representative set of samples and building up a training and testing system.

3.2.1.2 Histogram distribution and quantisation

A histogram represents the distribution of a set of univariate data. The range of the data is divided in each bin. Histograms calculates the occurrences of data into bins. The bin counts and column size can be viewed as a density estimate of data distribution [Sircombe, 2000]. The histogram also represents the underlying probability density distribution which the absolutely continuous probability density function $p(x)$ for given function $f(x)$ which x in limit $[a, b]$ is shown in Equation 3.2 [Parzen, 1962]

$$\boldsymbol{p}(x) = \int_a^v \boldsymbol{f}(x)dx \tag{3.2}$$

We use the Shannon [2001]'s entropy h to measure the uncertainty of the probability density function $p(x)$ which is

$$h = -\sum_{i=1}^{M} \boldsymbol{p}(x)_i \, log_2 \boldsymbol{p}(x), i \in (1...M) \tag{3.3}$$

The p is previously described as the probability density function. A large value of entropy h means the distribution of histogram is smooth and the small entropy h means the distribution is spiky. Since we want to get the distances between languages, it is important that the distances can be vary as much as possible, which means we need a large entropy for our tasks. One issues is how to define those intervals when they are not pre-defined, in the other word, histogram quantisation.

One of the earliest guidelines of histogram quantisation was proposed by Sturges

[1926] in which the bin width, w, is proposed as:

$$w = \frac{\Delta}{1 + log_2 n}, \tag{3.4}$$

where n is the number of points in the dataset and Δ is the range of the data. However, Scott [1979] claimed that Sturges [1926]'s work tends to over-smooth as w is going to be small when applied to large dataset.

$$w = 3.49\sigma n^{\frac{-1}{3}}, \tag{3.5}$$

Alternatively, Freedman and Diaconis [1981] used the interquartile range r of the data n for width quantisation. They claim this method is less sensitive to outliers:

$$w = 2\frac{r}{\sqrt[3]{n}}, \tag{3.6}$$

Since we have 254 languages the in text language identification database and outliers are not especially problematic, we are going to use Scott [1979]'s equation 3.5 to measure the bin width.

In this thesis, we compare the language distances between different n-grams and penalties. It is inappropriate to define a unique w value for all distances matrices. So, in order to compare the distance distribution by the same axis value, we use the bin width $\Delta = w/\sigma$ instead of w, where σ is the standard deviation of the histogram. It means the distance shown in the diagram is d/σ, where d is the distance between languages. Thus, the entropy of histograms can be compared by the same bin width in all cases (otherwise the bin width of histogram can be vary in each experiment). This also avoids the problem that entropy 3.3 can also be sensitive to bin width.

3.2.1.3 Cross validation

To evaluate how well a particular model works, the model needs to be tested on a new dataset which has not been seen before. In order to use the dataset efficiently,

the cross-validation method uses each part of the training dataset for the testing. In our case, we use the 10-fold cross validation. Figure 3.3 describes the 10-fold cross-validation process. We split the dataset D (text in this section but can also

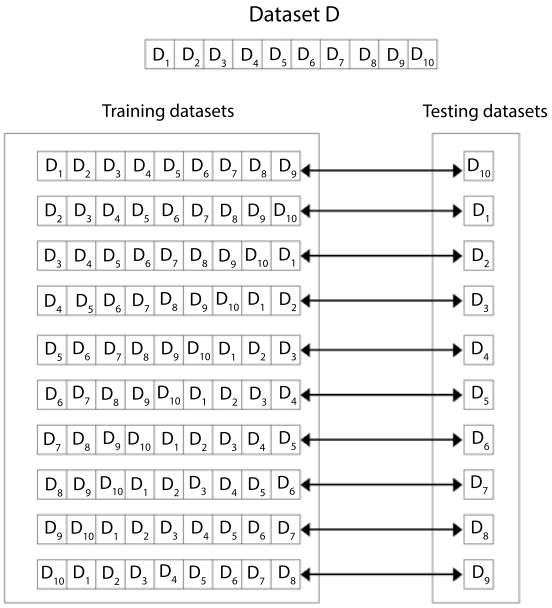


Figure 3.3: Cross validation process.

.

be audio or video) into 10 parts which are $\{D_1, D_2, D_3, D_4, D_5, D_6, D_7, D_8, D_9, D_{10}\}$ and we alternately "hold-out" one of the subsets for testing. For test segmentation, we have decided whether to split the main dataset at character boundaries, word boundaries, sentence boundaries or something larger. All of the methods need an accurate measurement of character n-grams. So, while splits of character boundaries, allowing for near-equal size subsets, they will introduce character n-gram errors. Sentence-level segmentation reduces the character n-gram errors but leads to the subset size potentially varying by the difference in sentence lengths. For this reason, in this work, we have chosen to split at word boundaries.

3.2.2 Phylogenetic tree clustering

Benedetto et al. [2002] first proposes a new language identification method. They state that, instead of obtaining very precise meanings of strings, language identific-

ation techniques are concerned more about the difference between languages. His idea is not only available in language phylogenetics but also works for other areas such as authorship attribution, music classification, image identification and optical character recognition. They assume that the distances between compressed pairs of sequences are related to the real semantic differences of the sequences.

This idea is initially from Kolmogorov complexity method which is originally used for analysing the shortest length of program for computing a given task. The main idea of Kolmogorov complexity is that, when there is a task to compute a string $\boldsymbol{a}$, the shortest length of this program is $k(\boldsymbol{a})$. Suppose computing string $\boldsymbol{a}$ when string $\boldsymbol{b}$ is appended into the program, the Kolmogorov complexity could be presented as $k(\boldsymbol{a}|\boldsymbol{b})$ and the distance $d(\boldsymbol{a},\boldsymbol{b})$ between the string $\boldsymbol{a}$ and the string $\boldsymbol{b}$ is defined by equation 3.7.

$$d_{\boldsymbol{ab}} = \frac{k(\boldsymbol{a}|\boldsymbol{b}) + k(\boldsymbol{b}|\boldsymbol{a})}{k(\boldsymbol{ab})} \tag{3.7}$$

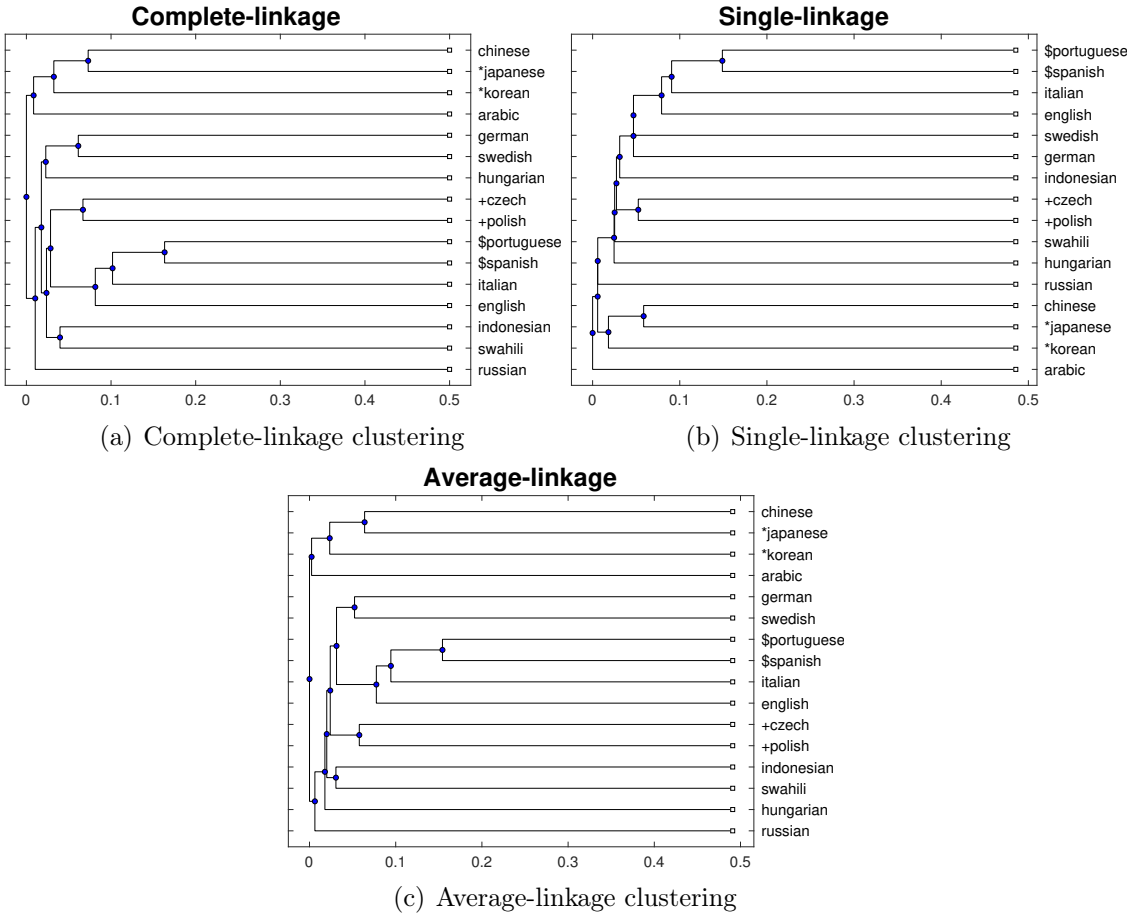
Benedetto et al. [2002] uses a function so called the relative entropy to evaluate the differences between the languages A and the language B by compressing the string $\boldsymbol{a}$ from the language A and the string $\boldsymbol{b}$ from the language B. Equation 3.8 shows the distances S_{AB} which is the so called relative entropy by Benedetto et al. [2002]. $\boldsymbol{a_s}$ is a substring of $\boldsymbol{a}$ and $\boldsymbol{b_s}$ is a substring of $\boldsymbol{b}$. $\Delta\boldsymbol{ab_s} = L(\boldsymbol{ab_s}) - L(\boldsymbol{a})$ which $L(\boldsymbol{a})$ means the length in bits of the zipped string $\boldsymbol{a}$.

$$S_{AB} = \frac{\Delta\boldsymbol{ab_s} - \Delta\boldsymbol{bb_s}}{\mid \boldsymbol{b} \mid}, \tag{3.8}$$

According to Benedetto et al. [2002]'s theory, it is possible to use an evolution tree to describe the relationships between the languages by zipping. Thus we applied the phylogenetic tree clustering to evaluate the results.

As a kind of hierarchy clustering, which tries to build a hierarchy of clusters, the phylogenetic tree computes the distances between clusters and nodes: the complete-linkage clustering, the single-linkage clustering and the average-linkage clustering.

The complete-linkage clustering merges two clusters with the smallest maximum pairwise distances, the single-linkage clustering merges two clusters with the smallest minimum pairwise distances and the average-linkage clustering merges two clusters with the smallest average pairwise distances. The complete-linkage clustering is more sensitive to the outlier while the single-linkage clustering might cause a long chain of clusters. The average-linkage clustering is the compromise of these two clustering[Tsvetovat and Kouznetsov, 2011]. Figure 3.4 shows examples which is



(a) Complete-linkage clustering

(b) Single-linkage clustering

(c) Average-linkage clustering

Figure 3.4: Diagrams display the differences between complete-linkage clustering, single-linkage clustering and average-linkage clustering. Figure 3.4(a) shows the tree structure built by complete-linkage clustering, Figure 3.4(b) shows the tree structure built by single-linkage clustering and Figure 3.4(c) shows the tree structure built by average-linkage clustering.

displayed by tree structures and the tree structures are built by complete-linkage clustering, single-linkage clustering and average-linkage clustering. Figure 3.4(a)

shows the tree structure built by complete-linkage clustering, Figure 3.4(b) shows the tree structure built by single-linkage clustering and Figure 3.4(c) shows the tree structure built by average-linkage clustering. According to the linguistic language tree which is previously described in Figure 2.7, it is obvious that the complete-linkage clustering shows more language structures and performs a better grouping because all Indo-Hittite languages are grouped under one subtree. Thus, we use complete-linkage clustering instead of the other two.

3.2.2.1 Hypothesis Test

To evaluate whether the language distance distributions are significantly different, we need to use the hypothesis test. By using the hypothesis test, we can understand whether these differences occur more often than chance. The hypothesis test uses a null hypothesis H_0 which means there is no significant difference between two samples. The probability p value shows that the result is possible to occur if H_0 were true. The null hypothesis of H_0 will be rejected if p is lower than the significant level of $p < 0.05$, 0.01, 0.005 or 0.001 [Manning and Schütze, 1999].

The t-test is one of the most common hypothesis tests which looks at the mean of two independent normal populations. The equation of calculating the value of the t-test is shown in Equation 3.9.

$$t = \frac{\overline{x}_1 - \overline{x}_2}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}} \tag{3.9}$$

which $\overline{x}_1$ is the mean of the first data set and $\overline{x}_1$ is the mean of the second data set. S_1 is the sample variance of the first data set and S_2 is the sample variance of the second data set. n_1 is the sample size of the first data set and n_2 is the sample size of the second data set. Thus, the null hypothesis H_0 is the mean of the two data set are equal and the alternative hypothesis H_a is the mean of the two data set are not equal.

3.2.3 Results

In this section, we use histograms to display language distance variation. The method used for determining bin width for histogram is previously detailed in Section 3.2.1.2. The distances are calculated via Cavnar and Trenkle [1994]'s n-gram model with 10-fold cross validation which was previously explained in Section 3.2.1.3. This task uses the UNDHR dataset which is listed in appendix A. The predefined maximum "out-of-place" value, which we called "penalty" here has a default value of 400 by Cavnar and Trenkle [1994]. Note here we are going to vary the default penalty parameter of 400 later.

Table 3.3: Entropy(top) and accuracy(bottom) values with histogram binwidth = 0.13.

	Entropy							
Penalty value	1	5	10	50	100	400	500	1000
Gram=1	4.70	4.41	4.31	3.93	3.90	3.89	3.90	3.89
Gram=2	4.61	4.71	4.85	4.99	4.84	4.73	4.72	4.71
Gram=3	4.38	4.38	4.37	4.50	4.33	4.34	4.33	4.34
Gram=4	4.44	4.45	4.44	4.60	4.34	4.32	4.29	4.33
Gram=5	4.44	4.43	4.46	4.67	4.41	4.30	4.30	4.24
	Accuracy value							
Gram=1	0.04	0.62	0.77	0.79	0.74	0.65	0.65	0.65
Gram=2	0.00	0.00	0.00	0.42	0.62	0.82	0.82	0.84
Gram=3	0.00	0.00	0.00	0.06	1.00	1.00	1.00	1.00
Gram=4	0.00	0.00	0.00	0.03	0.99	1.00	1.00	1.00
Gram=5	0.00	0.00	0.00	0.02	0.98	1.00	1.00	1.00

Table 3.3 shows the recognition accuracy and entropy of uni-gram, bi-gram, tri-gram, quad-gram, five-gram results with different bin widths. To compare histograms with different penalties and n-grams, we test the same bin width for all language distances. Those bin widths are all calculated by function 3.5. And also, to compare the histogram distributions and entropies for all pairs of n-grams and penalties, it is necessary to use the same bin width and counts the number of occurrences. According to Scott [1979] which is explained in Section 3.2.1.2, the optimal bin width for this task is 0.13.

Figure 3.5 shows examples of histogram distributions with the maximum and

minimum entropy. According to Section 3.2.1.2, the fixed x-axis value is $3.49n^{\frac{-1}{3}}$ by using the same bin width $w = 0.13$. Figure 3.5 shows, the x-axis, the distance values of histogram d is divided by the standard deviation σ (d/σ). The y-axis shows the probability density in each bin for Cavnar and Trenkle [1994]'s distances. The probability density $pd_i = \frac{h_i}{n}$ which h_i is the count in the i-th bin and n is the size of language distance matrix (the total count). The yellow curve shows the histogram distribution of the lowest entropy in TLID n-gram results and the blue curve shows the histogram distribution of the highest entropy. The lowest entropy has a spiky distribution and the highest entropy has a smooth distribution. By testing the probability of the null hypothesis of the highest and the lowest distance matrix, the p value of the t-test is 0 which rejects the null hypothesis H_0 that there is no difference between the means. So, we can say that the distribution of the highest entropy and the lowest entropy are significantly different. Thus, as the high entropy can provide more distances information about languages, we prefer to choose high entropy with high accuracy result.

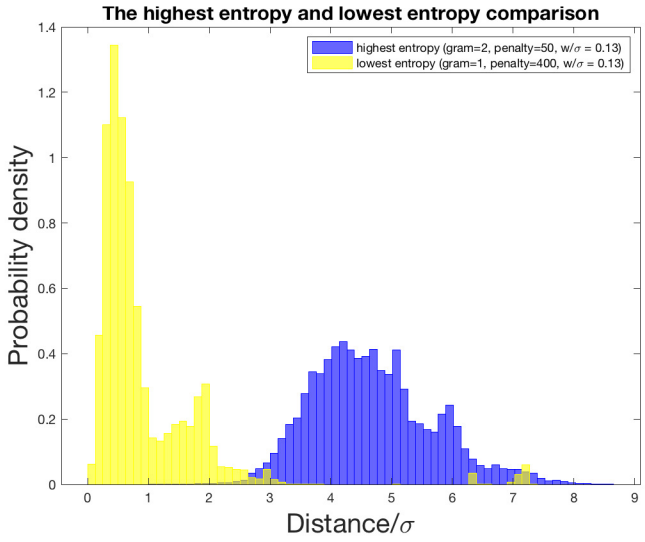


Figure 3.5: histogram distribution for highest and lowest entropy of language distances.

3.2.4 Conclusion

Table 3.3 tells the dramatic story of why this method is the method of choice for TLID - high accuracy with high entropy. We tested different penalty values on our model which are shown in Table 3.3. We found that, high penalty values are needed for high accuracy. However, if we need to use the distance to provide some subsequent language distances, then we want high entropy.

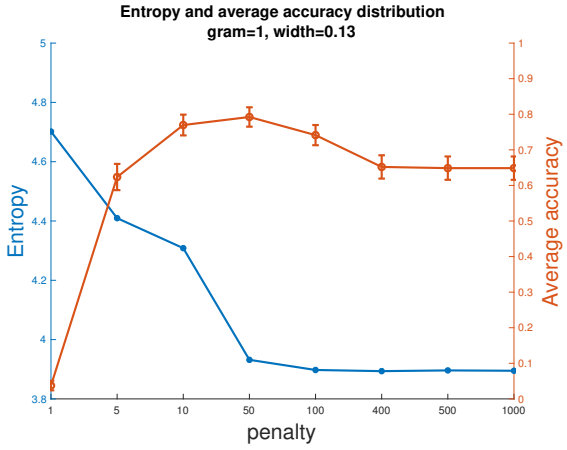
And also, given that the maximum penalty p is a parameter of the method, we might expect a graph showing how the accuracy and entropy of distance vary with p. Figure 3.6 shows the entropies and accuracies in each n-gram and varied by penalties (the histogram results for each penalty and n-grams are listed in Appendix B). We find in Figure 3.6(b), 3.6(b), 3.6(c), 3.6(d) and 3.6(e), the accuracies of n-gram increase with the penalties. For unigram results in Figure 3.6(a), we see the entropies in penalty 50, 100, 400, 500 and 1000 are similar because the unigram does not have too many n-grams. For example, English has 26 characters, so once the penalty is over 26, the penalty cannot impact on the language distance distributions.

To compare with zipping results in Section 3.3, and also to cope with ALID dataset, we extract 16 languages and build a colormap and a dendrogram to describe the language relationships. In ALID, the number of languages is 21 which are listed in Table 2.8. However, in the text database of UNDHR[2], Cantonese is written as Mandarin. Tamil, Hindi, Farsi, Vietnamese are printed as pictures which cannot be transformed into Unicode text files. In that case, we only describe 16 languages distances relationships in TLID. Appendix C lists the color map and language tree for each n-gram model and penalty. The colormap 3.7(a) displays the color density of distances. It displays the data as an image that uses the full range of RGB colors. Based on the linguistic language trees in Figure 2.7 and Figure 2.8, we can define three language subsets - Spanish and Portuguese, Korean and Japanese, Czech and Polish. In the colormaps, we denote Spanish and Portuguese in pink, Korean and

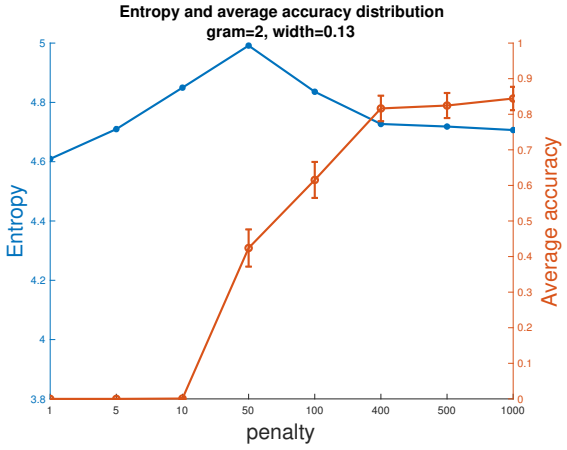
[2]https://www.ohchr.org/EN/UDHR/Pages/SearchByLang.aspx

Japanese in blue and Czech and Polish in red. In the dendrogram, we denote Spanish and Portuguese as symbol "$", Korean and Japanese as symbol "∗" and Czech and Polish as symbol "+".

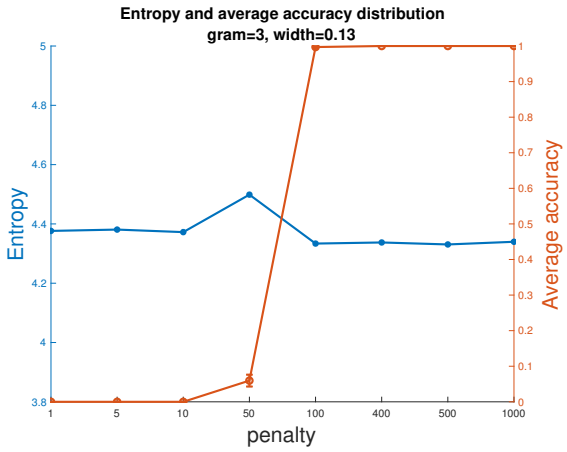
According to Figure 3.6, we can see the highest accuracy with the highest entropy is the trigram with 100 penalty, whose colormap and language tree are shown in Figure 3.7. Figure 3.7(a) tells the languages are all close to themselves which corresponds to the accuracy in Table 3.3. Comparing with Figure 2.8, we can find Catalan is close to Spanish, Czech is close to Polish. The colormap cannot tell that Japanese and Korean have a close relationship. However, Figure 3.7(b) shows that Japanese is close to Mandarin and Korean is close to Japanese. This is because written Japanese contains Sino-Japanese vocabulary which is written as Chinese characters in the text. Additionally, although Swahili and Indonesian are not European languages, as they were influenced by Dutch and English, their alphabets are consist of Latin characters. In this case, it is not surprising that these two languages are close to European languages.



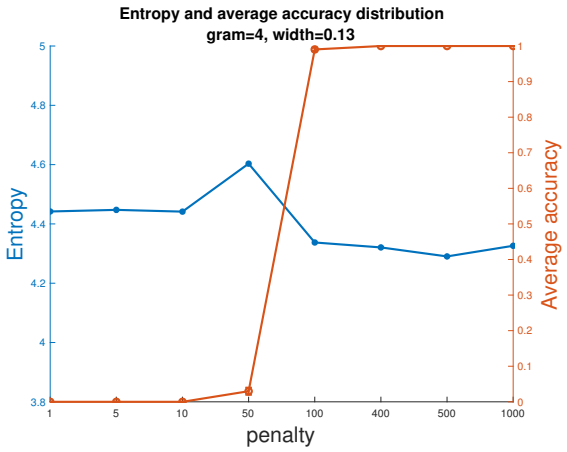
(a) Accuracy and entropy distribution for gram = 1



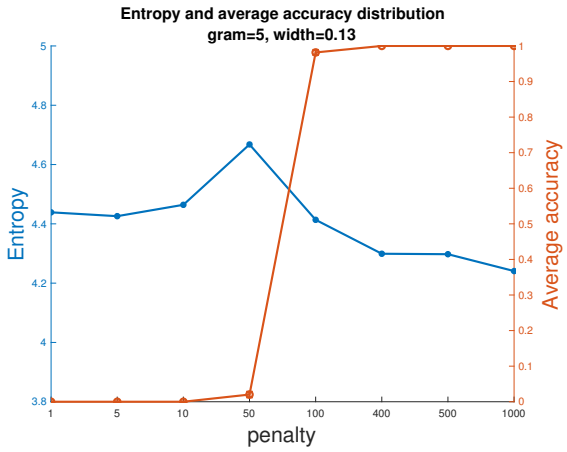
(b) Accuracy and entropy distribution for gram = 2



(c) Accuracy and entropy distribution for gram = 3

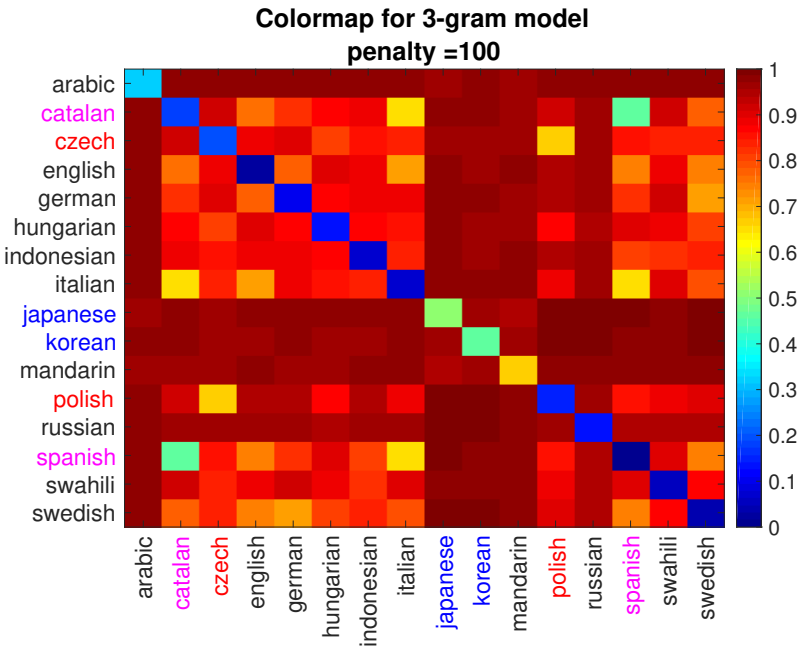


(d) Accuracy and entropy distribution for gram = 4

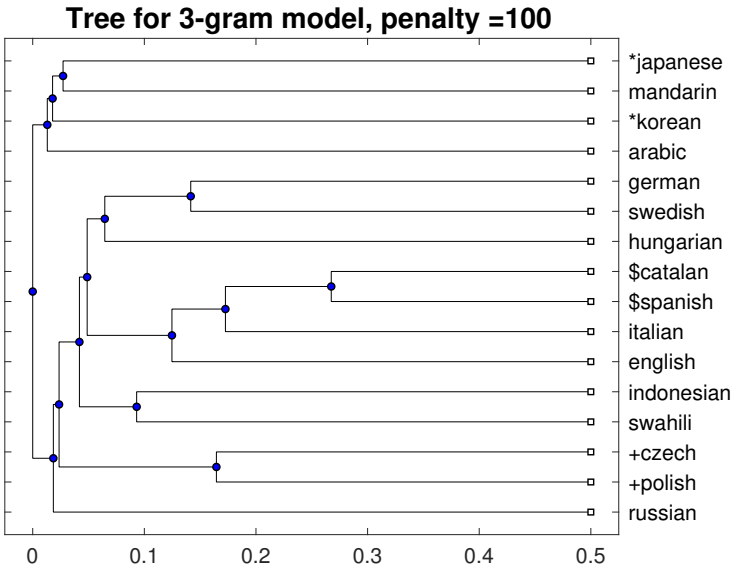


(e) Accuracy and entropy distribution for gram = 5

Figure 3.6: Accuracy and entropy distribution for n-grams. The x-axis is the penalty value. The left y-axis is the entropy value and the right y-axis is the accuracy value.



(a) Colormap of tri-gram



(b) dendrogram of tri-gram

Figure 3.7: The 16 UNDHR text language distances results of tri-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 100. Figure 3.7(a) shows the colormap of the language distance variations and Figure 3.7(b) shows the language tree which is built by the distances. The colour variation in Figure 3.7(a) shows the pairwise distances between languages.

3.3 Language distance calculated by compression

This section covers the computing of text language distances using compressors, which implement Benedetto et al. [2002]'s compression methods by using three compressors: zip, bzip and ppm. The detail of Benedetto et al. [2002]'s method is discussed in Section 3.2.2. The database we used in this project was UNDHR, encoded using Unicode. We have introduced all of the text languages we used for the zipping methods. Table 3.1 was previously mentioned in Section 3.1 and describes all of the languages we used in this project.

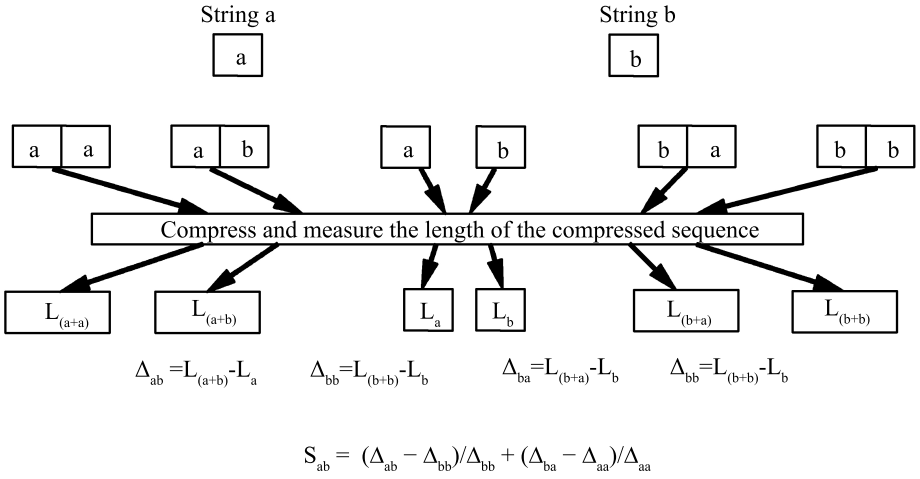


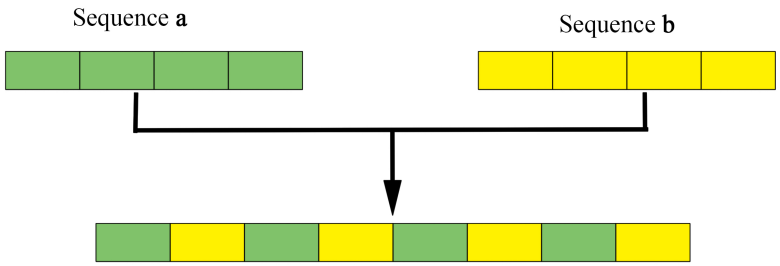
Figure 3.8: Compression on text and calculate the distance between two languages.

Figure 3.8 describes the language compression algorithm in detail. The basic requirement is to measure how compressible source $\boldsymbol{a}$ is given knowledge of itself and knowledge of $\boldsymbol{b}$. $L_{\boldsymbol{a}}$ is the length of compressed $\boldsymbol{a}$ and $L_{\boldsymbol{b}}$ is the length of compressed $\boldsymbol{b}$. To measure the distance of the sequence itself, we concatenate the string with itself; in this case, the length of the compressed sequence with itself should be $L_{(\boldsymbol{a}+\boldsymbol{a})}$ and $L_{(\boldsymbol{b}+\boldsymbol{b})}$. $L_{(\boldsymbol{a}+\boldsymbol{b})}$ is the length of compressed sequence $\boldsymbol{a}$ with sequence $\boldsymbol{b}$ and $L_{(\boldsymbol{b}+\boldsymbol{a})}$ is the length of compressed sequence $\boldsymbol{b}$ with sequence $\boldsymbol{a}$. The

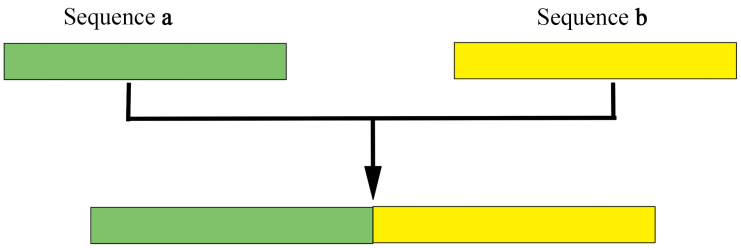
equation we used for measuring language distance is:

$$\begin{aligned} S_{\boldsymbol{ab}} &= (L_{(\boldsymbol{a}+\boldsymbol{b})} - min(L_{\boldsymbol{a}}, L_{\boldsymbol{b}})/max(L_{\boldsymbol{a}}, L_{\boldsymbol{b}}) \\ &\quad + (L_{(\boldsymbol{b}+\boldsymbol{a})} - min(L_{\boldsymbol{a}}, L_{\boldsymbol{b}}))/max(L_{\boldsymbol{a}}, L_{\boldsymbol{b}}), \end{aligned} \tag{3.10}$$

We also use the interleave and non-interleave methods on the text to prove whether the interleaved text would impact on our results. A simple explanation of the interleave and non-interleave methods is presented in Figure 3.9. According to Figure 3.9, the interleave methods chunk sequence $\boldsymbol{a}$ into $\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3, ..., \boldsymbol{a}_n$ and sequence b into $\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3, ..., \boldsymbol{b}_m$ then combine them as $\boldsymbol{a}_1, \boldsymbol{b}_1, \boldsymbol{a}_2, \boldsymbol{b}_2, \boldsymbol{b}_3, \boldsymbol{b}_3, ..., \boldsymbol{a}_n, \boldsymbol{b}_m$. The non-interleaved method combines sequence $\boldsymbol{a}$ with sequence $\boldsymbol{b}$ without chunking. To evaluate and describe the interleave and non-interleave results, we use colour maps and phylogenetic-like trees to show the distances between the text languages. Like TLID n-gram result, we also measure the entropy of the language distances for each zipping with interleaving and non-interleaving method. By looking at the entropy, we can find which method shows more language distance variations. Considering the accuracy and the entropy, we can conclude which method performs best.



(a) Interleave



(b) Non-interleave

Figure 3.9: Interleave and non-interleave methods

3.3.1 Language distance results via zip

3.3.1.1 Zip

In this section, we applied the "zip" command in Mac OS X, the default compression for which is deflate. Deflate compression is a lossless compression that combines LZ77 and Huffman encoding [Deutsch, 1996].

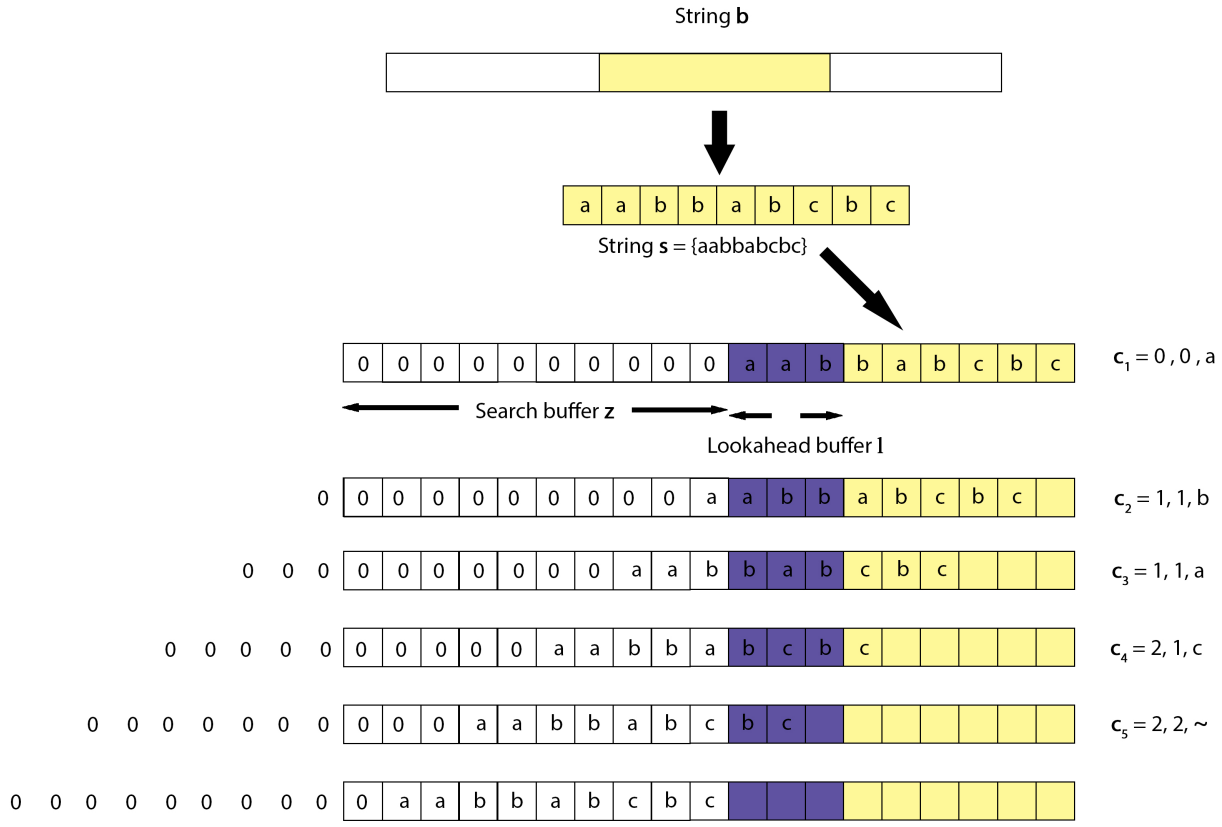


Figure 3.10: Lempel and Ziv compression [Ziv and Lempel, 1977]

LZ77 compression is proposed by Ziv and Lempel [1977] using code schemes, which map characters to bytes, to solve the data compression with limited knowledge of data source. Figure 3.10 shows a example of LZ77 process. The LZ77 replaces the repeated occurrences based on the previous uncompressed string and refers them to a fixed-length codeword $\boldsymbol{c}$, which is presented as $\boldsymbol{c}$ = {position, length, first non-matching symbol}. Position is the length between two repeated characters in $\boldsymbol{s}$ and

$\boldsymbol{l}$. Length is the offset that characters move into search buffer string $\boldsymbol{z}$. Figure 3.10 shows an example of the LZ77 process. String $\boldsymbol{s} = aabbabcbc$ is a part of a string $\boldsymbol{b}$. To compress string $\boldsymbol{s}$, LZ77 first sets an all-zero-string $\boldsymbol{z}$ with $length(\boldsymbol{z}) = 9$ as a search buffer and a lookahead buffer $\boldsymbol{l}$ with $length(\boldsymbol{l}) = 3$. An explanation of this process is given below.

1. The model checks the first character "a" in l and finds it does not match any character in $\boldsymbol{z}$. Then "a" is moved into the search buffer and the model defines the position = 0, length = 0 and the first symbol in $\boldsymbol{l}$ is "a". So the first codeword $\boldsymbol{c}_1 = (0, 0, a)$.

2. Now, the first character in $\boldsymbol{z}$ is still "a" and the same character is found in $\boldsymbol{l}$. So the "a" and the follow character "b" is moved into $\boldsymbol{z}$. In this case, $\boldsymbol{c}_2 = (1, 1, b)$.

3. The first character in Z is "b" and the same character is found in L and the distance between them is 1. So b and the following character "a" are moved into $\boldsymbol{z}$. In this case, $\boldsymbol{c}_3 = (1, 1, a)$.

4. The first character in $\boldsymbol{z}$ is "b" and the same character is found in $\boldsymbol{l}$. So the distance between them is 2. So "b" and the following character "c" are moved into $\boldsymbol{z}$. In this case, $\boldsymbol{c}_4 = (2, 1, c)$.

5. The first character in $\boldsymbol{z}$ is "b" and the same character is found in $\boldsymbol{l}$. Also, "bc" is found in $\boldsymbol{l}$. Considering the length is defined as the maximum length of repetition, the length = 2 and the distance between them is 2. So b and the following character "c" are moved into $\boldsymbol{z}$. The lookahead buffer is empty. Thus, $\boldsymbol{c}_5 = (2, 2, \emptyset)$.

6. The output string is $(0, 0, a), (1, 1, b), (1, 1, a), (2, 1, c), (2, 2, \emptyset)$.

3.3.1.2 Huffman coding

As one of the most famous data compression methods, Huffman coding provides an optimum binary coding. Huffman et al. [1952] referred to "message code" as the

symbols associated with a given message (e.g. string) and 'message length' as the time for the transmission message. Thus, the sum of probabilities $p(i)$ of n messages will be:

$$\sum_{i=1}^{n} p(i) = 1 \tag{3.11}$$

And for the average length of a message l_{per}, which also, as the number of coding digits of a message, is:

$$l_{per} = \sum_{i=1}^{N} pl(i) \tag{3.12}$$

Based on the above definitions, Huffman et al. [1952] suggested five limitations for constructing an optimum compression algorithm.

1. The identical coding digit sequence could not consist of two different messages.

2. When the start of a sequence is known, it is unnecessary to additionally point out where the begin and end of a message exist.

   In an optimum compressing code, to implement minimum redundancy, the shorter codes would be associated with the more probable messages. Thus, the relation between each probability of messages would be:

   $$p(1) \geq p(2) \geq \cdots \geq p(n-1) \geq p(n) \tag{3.13}$$

3. In corresponding to condition 2, the length of messages' relations are:

   $$l(1) \leq l(2) \leq \cdots \leq l(n-1) \leq l(n) \tag{3.14}$$

4. Despite the final digits, assuming there are d types of symbols used for coding,

if messages have the same code length $l(n)$, there are at least two, but no more than d types of symbols are alike.

5. Each possible sequence of $l(n) - 1$ digits should be used as a message code, otherwise its prefixes should be used as a message code.

When using three or more types of digits for messages, the optimum coding is similar to binary coding. A simple Huffman binary tree coding is shown in Figure 3.11. The Huffman binary tree satisfies all the restrictions mentioned earlier.

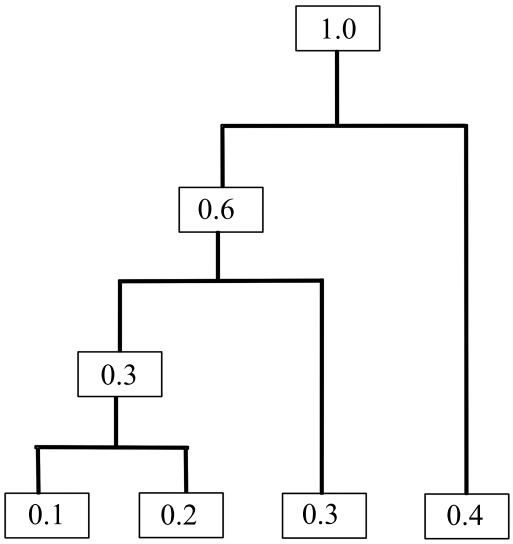


Figure 3.11: Simple Huffman coding

.

3.3.1.3 Results

This section describes the language distance distribution by using the colour map, phylogenetic tree and histogram distribution. The distances are calculated by the relative entropy which is described in Equation 3.10 and the compressor we use in this section is zip.

The description of phylogenetic tree is in Section 3.2.2 and the description of histogram distribution is in Section 3.2.1.2. Figure 3.12(a) and 3.12(b) show the

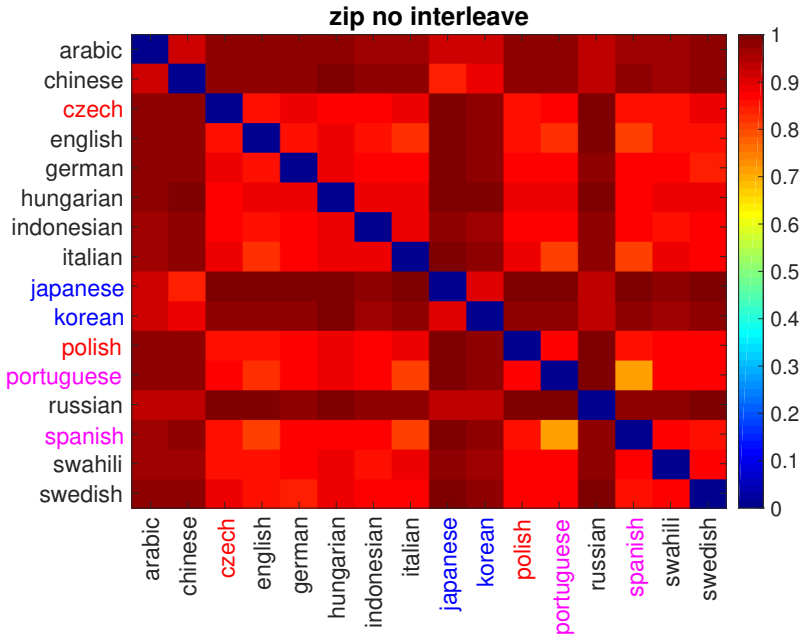
colour map of the languages distances. Figure 3.13(b) and 3.13(b) show the tree structure of language distances.

For compression results, a 0 following the name of the compressor denotes the interleaving status. For example, zip0, means non-interleaved string with the zip compressor and ppm1 means an interleaved string with the ppm compressor.

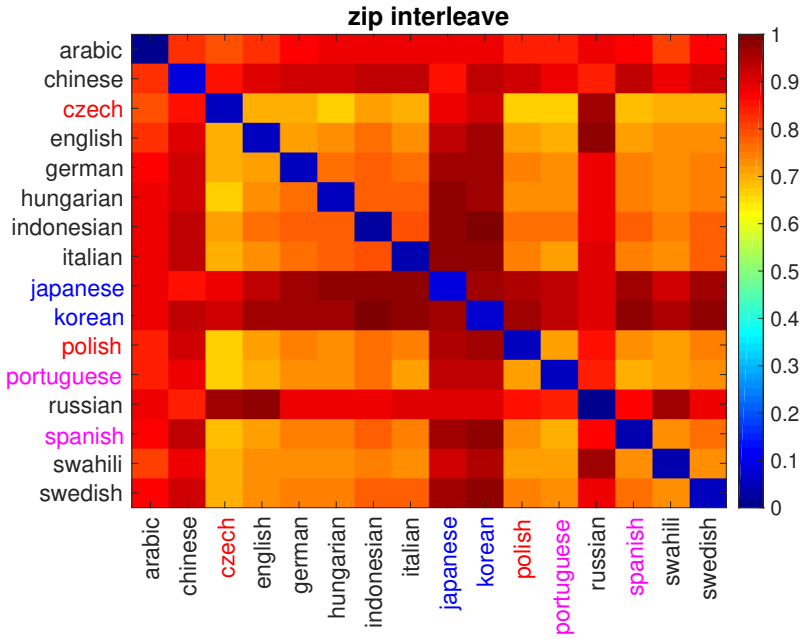
To illustrate the method we consider this distances generated via 3.13 using the zip algorithm 3.3.1. In Figure 3.12 we show density plots where the distance is colour-coded as in the right hand scale. We consider the cases: without interleaving and with interleaving case in Figure 3.12. The languages here are the UNDHR files described in Section 3.1.

There are some encouraging results in the non-interleave result (Figure 3.13(a)): Portuguese and Spanish, Czech and Polish are close. Languages that are isolated representatives of these trees, such as Arabic, Chinese, Japanese and Korean, are grouped together as they all have unique character system and far from all Indo-Hittite languages. Russian should be grouped as a part of Indo-Hittite language but in fact is not. According to the linguistic language background truth tree in Figure 2.7, we can see Russian is linguistically closed to Czech and Polish (although not under the same subtree). The reason is that Russian contains a lot of unique characters that can be viewed as different from other Indo-Hittite languages. The interleave result in Figure 3.13(b) shows a bad language grouping. Since the interleave change the occurrences of characters in the buff strings. Once the languages share part of characters, the interleave method destroys the structure of words and confused the classifier. This is why the interleave still can distinguish the Indo-Hittite languages from other languages.

Figure 3.14 shows the distribution of the pairwise distance between the 16 languages. If the distance between language i and j is $D_{ij}(i \leq j)$ then we show $1/2$ of D_{ij} since the distance matrix is symmetric and we do not want to calculate the language pair-wise distances twice. The distribution is presented by D_{ij}/σ, which σ is the standard deviation of D_{ij}. We use the entropy to summary the histogram



(a) without interleave

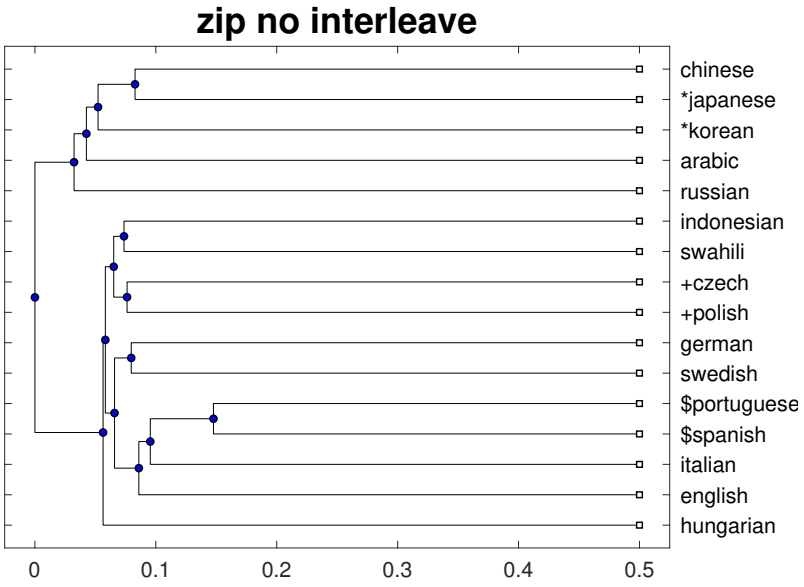


(b) with interleave

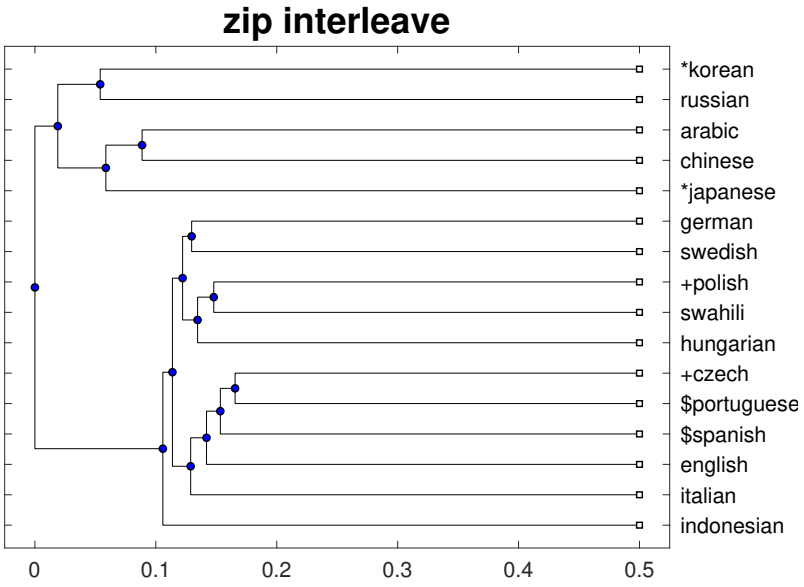
Figure 3.12: The 16 UNDHR text languages distances are computed by zip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. Figure 3.12(a) shows the non-interleaved result and Figure 3.12(b) shows the interleaved result.

distribution. The detail of the histogram and entropy is explained in Section 3.2.1.2.

We can see the interleave result gets a higher entropy rather than the non-interleave



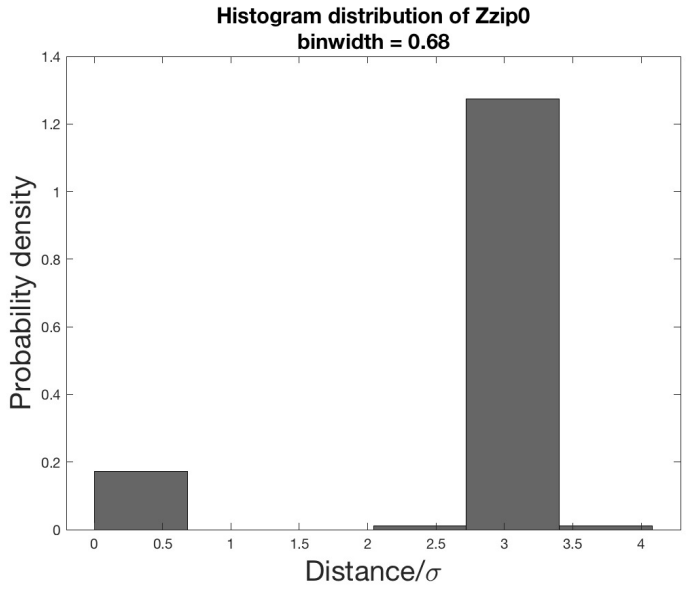
(a) without interleave



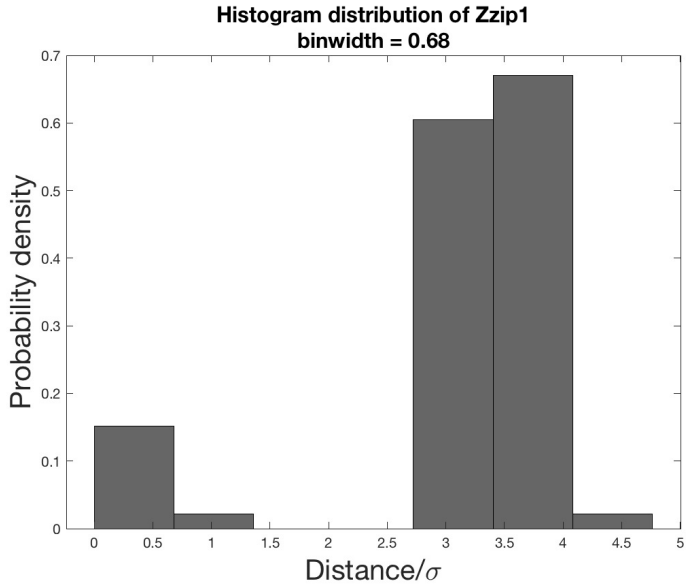
(b) with interleave

Figure 3.13: The 16 UNDHR text languages distances are computed by zip and the distance matrix is shown by tree structure. Figure 3.13(a) shows the non-interleave result and Figure 3.13(b) shows the interleave result. The length of branches between the points correspond with the distances between languages.

one. Although we want to get a higher entropy to show more distances variations, the language trees (in Figure 3.13) shows that the non-interleave method is a better choice rather than the interleave.



(a) without interleave



(b) with interleave

Figure 3.14: The 16 UNDHR text languages distances are computed by zip and the distance matrix are shown by histogram distributions. Figure 3.14(a) shows the non-interleave result and the entropy value of the histogram is 2.5. Figure 3.14(b) shows the interleave result and the entropy value of the histogram is 2.77.

3.3.2 Language distance results via bzip

3.3.2.1 Bzip

Bzip is a compression method that uses the Burrow-Wheeler transform followed by Huffman encoding (explained in Section 3.3.1.2). It claims that its compression performance is better than LZ77 and LZ78 and the accuracy is close to the PPM compressor [Seward, 1996].

3.3.2.2 BWT (Burrows-Wheeler Transform)

Burrows [1994] simply processes a block of text S as a single unit rather than as a sequential mechanism. Suppose string $\boldsymbol{s}$ = "$abcdc|$" has $n = 5$ characters with a "|" symbol which standards for the end of the string $\boldsymbol{s}$. By cycling shifts, rotations and sorting in lexicographical order for strings, the original and rotated string could form a $\boldsymbol{M} = n \times n$ matrix with contents as shown below:

Table 3.4: Burrow-Wheeler Transform.

Row	Rotation $\boldsymbol{M}$	Sorting $\boldsymbol{M}$
1	abcdc|	abcdc|
2	|abcdc	bcdc|**a**
2	c|abcd	cdc|a**b**
3	dc|abc	c|abc**d**
4	cdc|ab	dc|ab**c**
5	bcdc|a	|abcd**c**

Then, the last column of M is the transformed string $\boldsymbol{s}$ = "$|abdcc$".

Burrows [1994] argues that this kind of block text can easily be compressed by Huffman or arithmetic coding since it runs out the repeated characters after running MTF (Move to Front) transform and RLE (Run-Length encoding). Its performance could be comparable to statistical modelling techniques and the speeds obtained were as good as the Lempel-Ziv compressor.

3.3.2.3 MTF (Move to Front) and RLE (Run-Length Encoding)

MTF (Move to Front) applied permutation of the data into the index of alphabet dataset [Ryabko, 1980]. A 8-bit data string need a size of 255 identity permutation. It simply moves the symbol (which occurs in the data) into the front of permutation and count the displacement. Table 3.5 shows a simple example of MTF and the string is $\boldsymbol{s}$ = "*bannana*". For simple describe the process, the list of permutation symbols is the English alphabet.

Table 3.5: Move to Front.

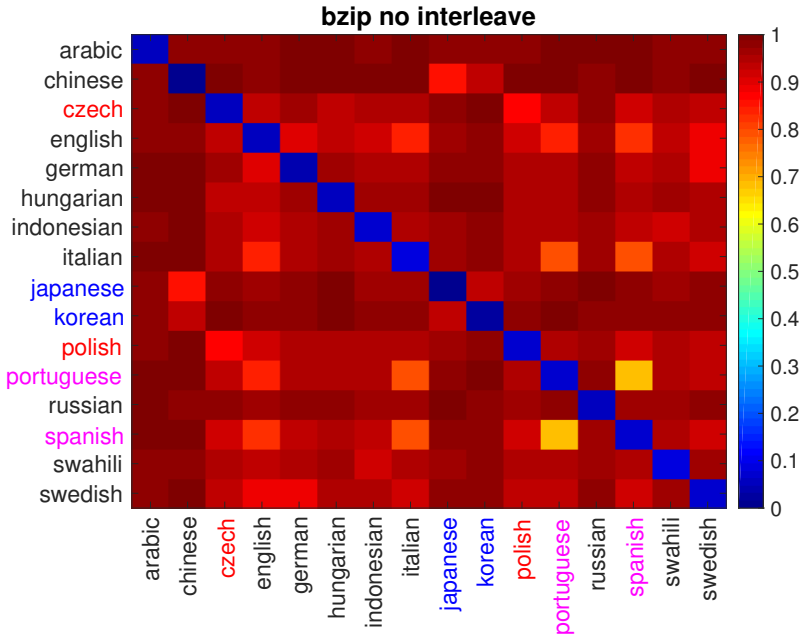
Row	Displacement	Permutation list
bannana	1	a**b**cdefghijklmnopqrstuvwxyz
b**a**nnana	1, 1	b**a**cdefghijklmnopqrstuvwxyz
ba**n**nana	1, 1, 13	bacdefghijklm**n**opqrstuvwxyz
ban**n**ana	1, 1, 13, 0	**n**bacdefghijklmopqrstuvwxyz
bann**a**na	1, 1, 13, 0, 2	nb**a**cdefghijklmopqrstuvwxyz
banna**n**a	1, 1, 13, 0, 2, 1	a**n**bcdefghijklmopqrstuvwxyz
bannan**a**	1, 1, 13, 0, 2, 1, 1	n**a**bcdefghijklmopqrstuvwxyz

The first step describes that the first character of string $\boldsymbol{s}$ is "b". To move "b" to the front of list, the displacement is 1. The other steps are followed by the first step and the string $\boldsymbol{s}$ is encoded into $1, 1, 13, 0, 2, 1, 1$.

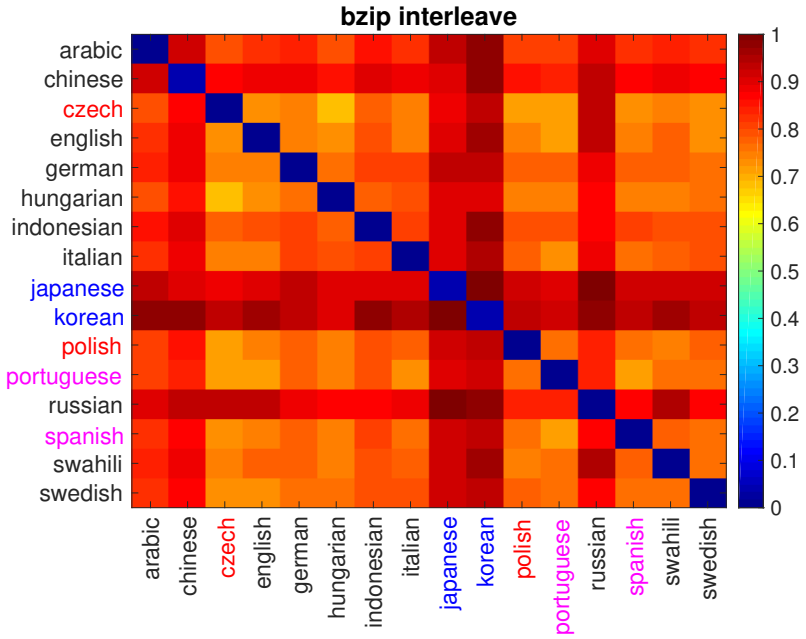
RLE (Run-length Encoding) simply calculates the occurrences for each characters [Robinson and Cherry, 1967]. It counts the transformed string $\boldsymbol{s} = \{1, 1, 13, 0, 2, 1, 1\}$ into $\boldsymbol{s} = \{21113101221\}$. The first 2 means character "1" occurs twice at the first time and the other numbers in the string $\boldsymbol{s}$ describe the same story as it. This method is helpful when the string contains many long repeated characters.

3.3.2.4 Results

This section describes the language distance distribution by using colour map, dendrogram and histogram distribution. The distances are calculated by the relative entropy which is described in Equation 3.10 and the compressor we use in this section is bzip.



(a) without interleave



(b) with interleave

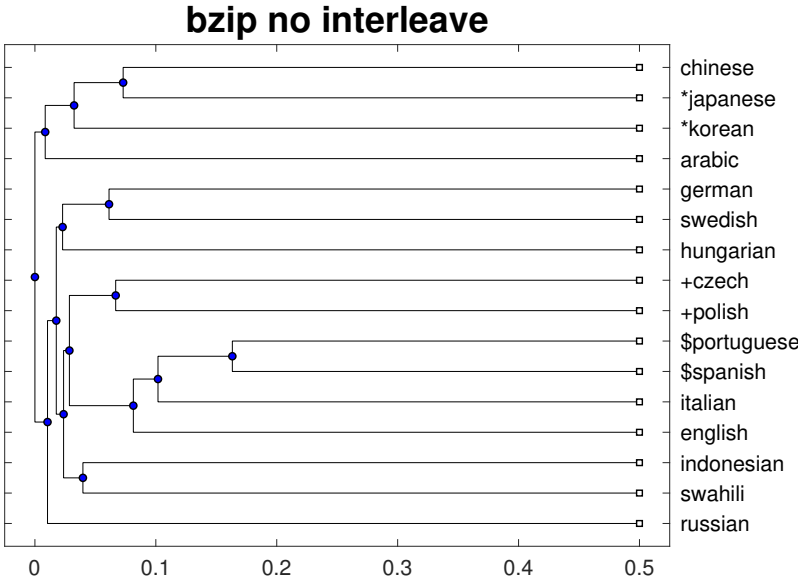
Figure 3.15: The 16 UNDHR text languages distances are computed by bzip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. Figure 3.15(a) shows the non-interleaved result and Figure 3.15(b) shows the interleaved result.

The description of dendrogram is in Section 3.2.2 and the description of histogram distribution is in Section 3.2.1.2. Figure 3.15(a) and 3.15(b) show the colour map

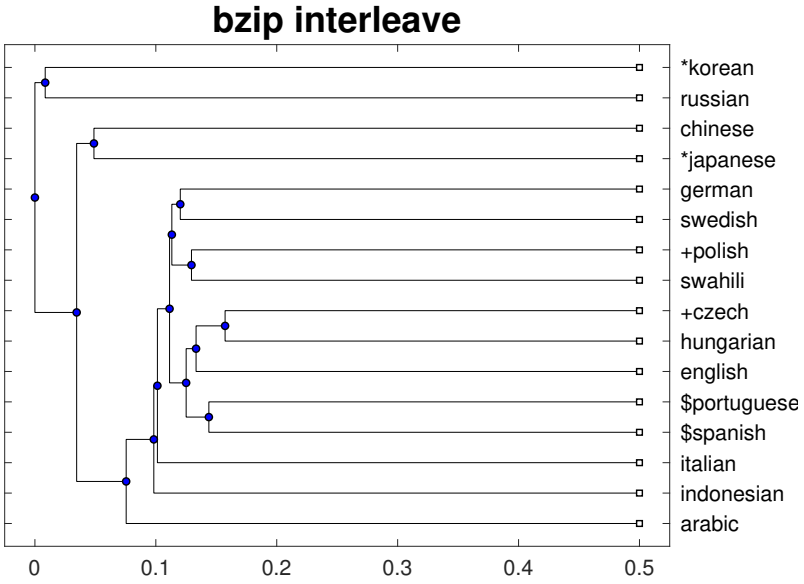
of the languages distances. Figure 3.16(b) and 3.16(b) show the dendrogram of language distances.

Figure 3.15 shows the colour map of the pairwise distances between the languages which is produced by the same methods as Figure 3.12 but with bzip as the compressor. Figure 3.15(b) shows more distance variation of language distances than 3.15(a) because the interleaving method shows more character variations in one buffer string. What is positive in both colour map is, the languages are all close to themselves. And also, in both Figure 3.15(a) and 3.15(b), Portuguese and Spanish is close. What is more, the language tree in Figure 3.16(a) shows that Czech and Polish, Japanese and Korean are also close. Russian shows a closer distance to the other Indo-Hittite language in the non-interleaving result, which shows that bzip performs better than zip. The interleaving result performs worse than the non-interleaving result is because that the interleave change the occurrences of characters in the buff strings. Once the languages share part of characters, the interleaving method destroys the structure of words and confused the classifier. This is why the interleaved method still can distinguish the Indo-Hittite languages from other languages.

Figure 3.17 shows the distribution of pairwise distance between the 16 languages. This diagram is produced by the same methods as Figure 3.14. We use the entropy to summarise the histogram distribution. The detail of the histogram and entropy is explained in Section 3.2.1.2. We can see the interleaving result gets a higher entropy rather than the non-interleave one. Although we want to get a higher entropy to show more distances variations, the language trees (in Figure 3.16) tells that the non-interleaving result is a better choice rather than the interleave.

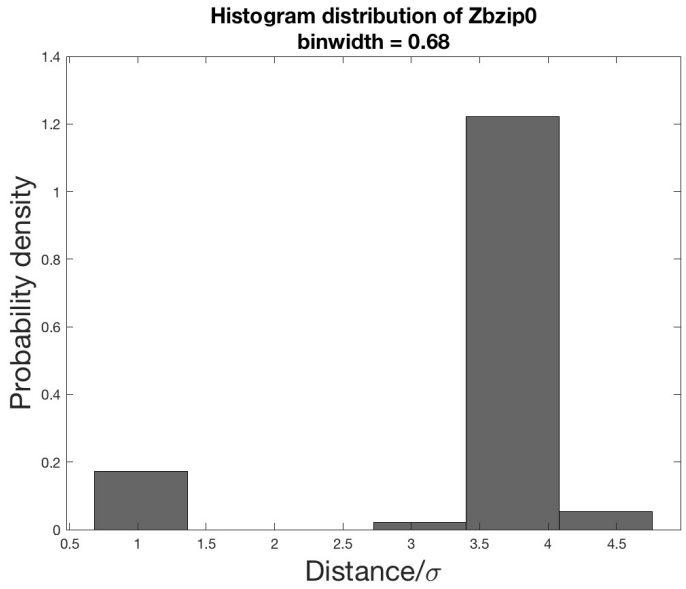


(a) without interleave

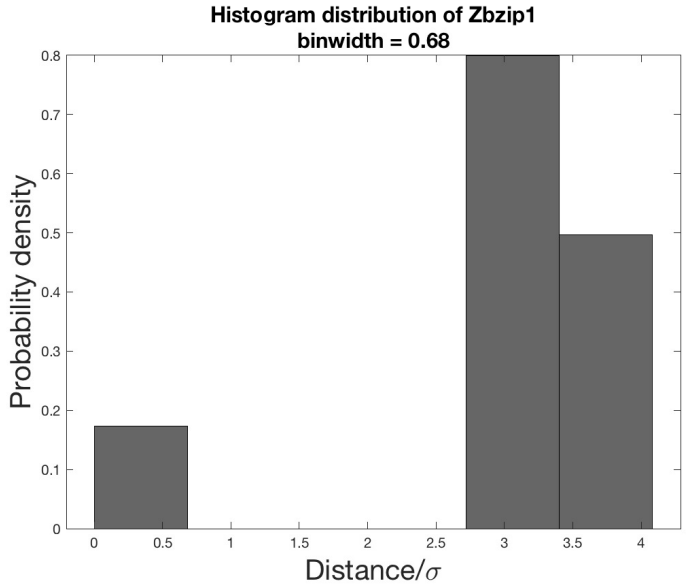


(b) with interleave

Figure 3.16: The 16 UNDHR text languages distances are computed by bzip and the distance matrix is shown by dendrogram. Figure 3.16(a) shows the non-interleaved result and Figure 3.16(b) shows the interleaved result. The length of branches between the points correspond with the distances between languages.



(a) without interleave



(b) with interleave

Figure 3.17: The 16 UNDHR text languages distances are computed by bzip and the distance matrix are shown by histogram distributions. Figure 3.17(a) shows the non-interleaving result and the entropy value of the histogram is 2.5. Figure 3.17(b) shows the interleaving result and the entropy value of the histogram is 2.54.

3.3.3 Language distance results via PPM

3.3.3.1 PPM (Prediction by partial matching)

Prediction by partial matching (PPM) uses 'adaptive coding' to dynamically update the model for compression. To solve the inefficient coding of the high-order Markov model, Cleary and Witten [1984] introduces 'partial match' so that the high-order Markov model can count the frequency faster with high compression quality.

Figure 3.18 shows a simple Markov chain for prediction. State A, B, C are codes and P is the probability of state transmission.

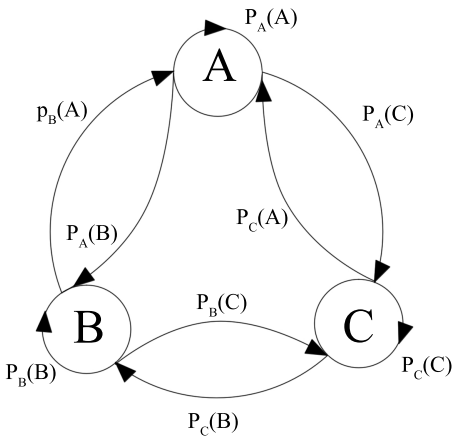


Figure 3.18: Markov chain

PPM uses a prepared coding alphabet that denotes all probabilities for each code. The coding alphabet can be ASCII, Unicode or other codes. For example, the string "How are yo" into the model and, assuming the order is a trigram, it finds that the highest probability after "yo" is "u". PPM uses the dynamic length order of the Markov model, which means the order of the Markov chain can be bigram, trigram or more. This can avoid the long length order with large numbers of infrequent predictions.

To deal with characters that have not been seen before, PPM provides an 'escape mechanism' that is partly similar to arithmetic coding methods. For example, the

number of a known character x occurs after 'yo' in the context is n times. PPM calculates the number of times m that 'yo' has occurred, then the probability of x is $p(x) = \frac{n}{m+1}$. If there are three characters $x = (a, b, c)$ that have not occurred before, the probability of all seen characters k should be $p(k) = \sum p(k)$. Thus, the probability of $s = 1 - \sum p(k) = \frac{1}{m+1}$. Suppose the size of the coding alphabet is n_a and the known characters are n_k, then the probability of each new character should be $\frac{1}{m+1} \times \frac{1}{n_a - n_k}$.

3.3.3.2 Arithmetic encoding

Witten et al. [1987] introduced arithmetic encoding into data compression. Theoretically, Huffman coding only has the "minimum redundancy" (best performance) under the circumstance in which all symbol probabilities are integral powers of $\frac{1}{2}$. The worst performance for Huffman coding would be when one symbol has a probability approaching unity, which, generally, sophisticated models predict.

The model predefines the probability to each symbol. This step can be done by counting frequencies of each symbol in a sample of text to be transmitted. Here is a symbol example.

Suppose there is a small alphabet in which $A = \{m, n, o, p, q, s\}$, then a fixed model with probabilities is shown in Table 3.6.

Symbol	Probability	Range
m	0.2	$[0, 0.2)$
n	0.3	$[0.2, 0.5)$
o	0.1	$[0.5, 0.6)$
p	0.2	$[0.6, 0.8)$
q	0.1	$[0.8, 0.9)$
s	0.1	$[0.9, 1.0)$

Table 3.6: Arithmetic encoding

When inputting a message 'S = "nmoos"', we model the rescale interval at each stage. Figure 3.19 presents the arithmetic coding process.

When finding the first symbol n, the model narrows it to $[0.2, 0.5)$, which means

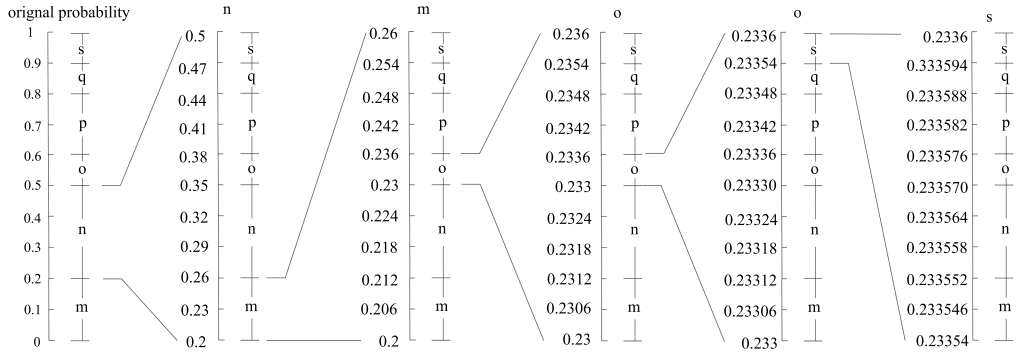


Figure 3.19: Arithmetic coding process with interval scaled at each stage

.

the rang of $[0.2, 0.5)$ indicates the symbol n. The second symbol m narrow the range into the first one-fifth of it and the previous range is 0.3 and the one-fifth of 0.3 is 0.06. Then the range turns into $[0.2, 0.26)$. Proceeding in this way, the final value of the range is $[0.23354, 0.2335)$.

3.3.3.3 Results

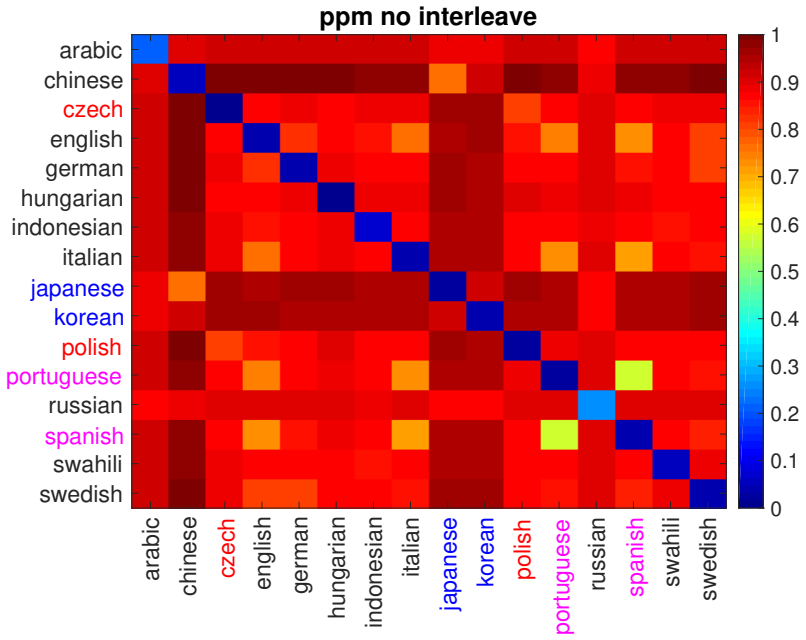
This section describes the language distance distribution by using colour maps, phylogenetic trees and histogram distributions. The distances are calculated by the relative entropy which is described in Equation 3.10 and the compressor we use in this section is ppm.

The description of phylogenetic tree is in Section 3.2.2 and the description of histogram distribution is in Section 3.2.1.2. Figure 3.20(a) and 3.20(b) show the colour map of the languages distances. Figure 3.21(b) and 3.21(b) show the tree structure of language distances.

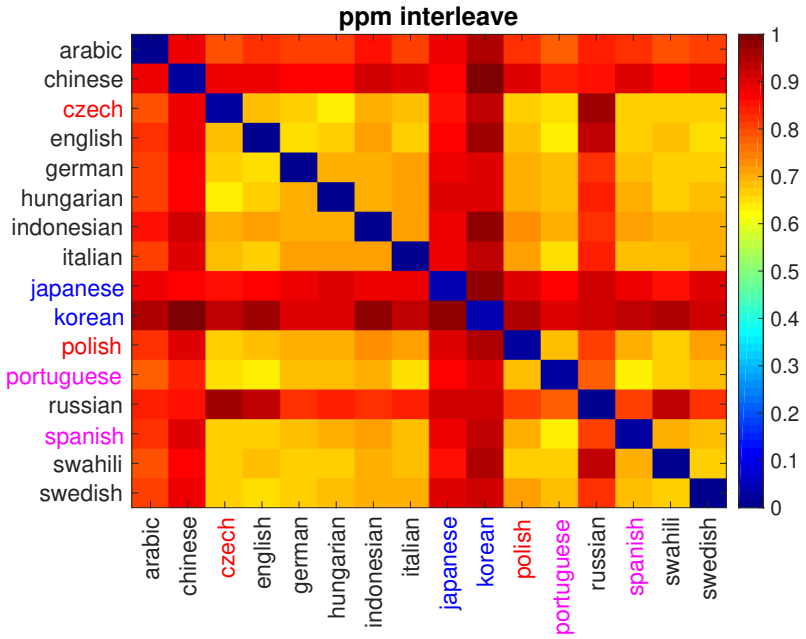
To illustrate the method we consider the distances generated via 3.21 using the ppm algorithm 3.3.3. In Figure 3.20(b) we show density plots where the distance is colour-coded as in the right hand scale. We consider the cases: without interleaving and with interleaving case in Figure 3.20. The languages here are the UNDHR files described in Section 3.1.

Figure 3.20 shows the colour map of the pairwise distances between the languages which is produced by the same methods as Figure 3.12 but with ppm as the compressor. We can find that both interleave and non-interleaving result show all languages are close to themselves. The distance between Portuguese and Spanish, Czech and Polish are significantly close to each other. Like zip and bzip results, the non-interleave ppm result shows that Japanese is close to Chinese rather than Korean for sharing Chinese characters in the writing system. But comparing to zip and bzip, the interleave ppm result shows fewer distance variations. Which means that ppm is heavily impacted by the interleaving because there is a rapid change in the context of the string as in $[A|B]$ leads to non-optimal compression.

Figure 3.22 shows the distribution of the pairwise distance between the 16 languages. This diagram is produced by the same methods as Figure 3.14. We use the entropy to summary the histogram distribution. The detail of the histogram and entropy is explained in Section 3.2.1.2. To compare different language distances, we



(a) without interleave

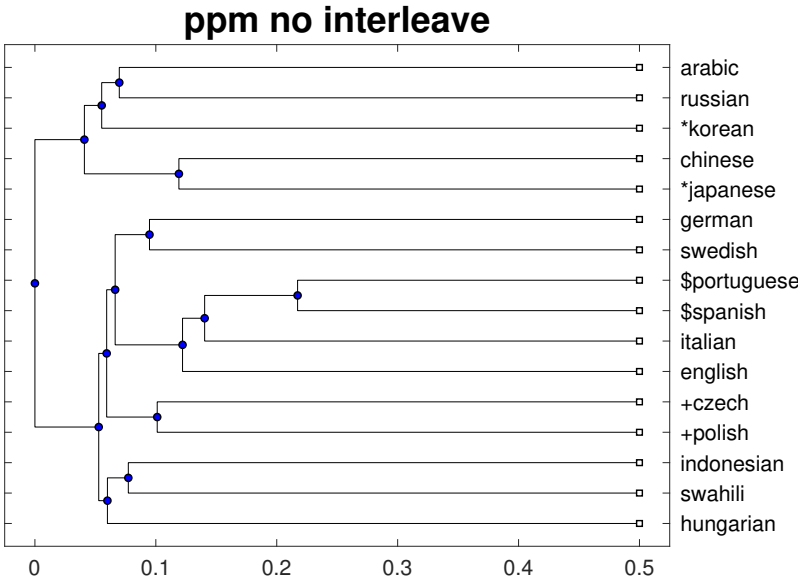


(b) with interleave

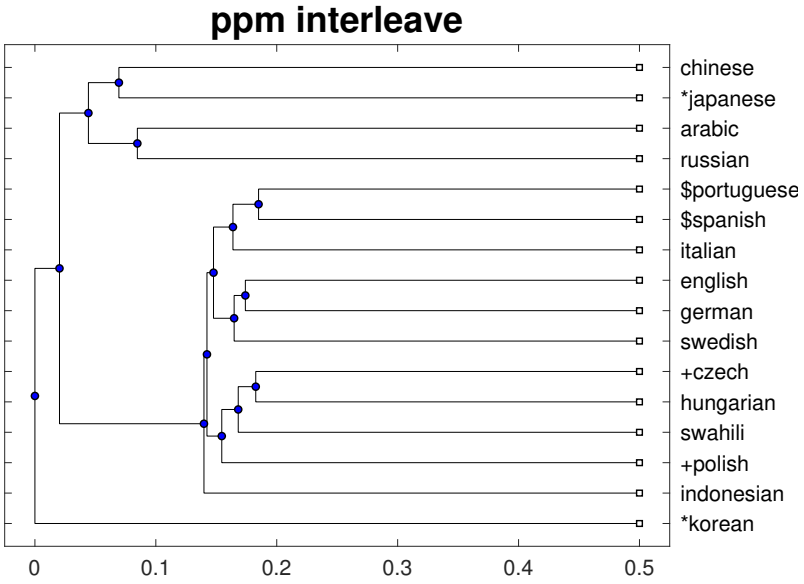
Figure 3.20: The 16 UNDHR text languages distances are computed by ppm and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. Figure 3.20(a) shows the non-interleaved result and Figure 3.20(b) shows the interleaved result.

need a fixed bin width which calculated by the same method as n-gram (see Section 3.2.1.2). We can see the interleaving result gets a higher entropy rather than the

non-interleave one. However, as we previous study in Appendix C, we can find that once the classification accuracy is 100%, the entropy only show the internal structure (whether the distances are further or closer) but does not change the relationships between the languages (like Chinese is always close to Japanese, German is always close to Swedish in tri-gram with 100, 400, 500 and 1000 penalty). For different methods like interleaved and non-interleaved, since entropy is impacted by different factors, it can only tell the distance variations between the languages instead of which result is better. Comparing with linguistic language tree in Figure 2.7 and Figure 2.8, we can find for both interleave and non-interleaving result, Portuguese and Spanish are under the same sub-tree, Korean and Japanese are also close to each other. The reason why Japanese is next to Chinese is explained in Section 3.2.4. However, the non-interleaving result shows that Czech and Polish are close, which is better than the interleaving result. Thus, although we want to get a higher entropy to show more distances variations, it shows that the ppm with the non-interleaving result is a better choice rather than the interleave one.

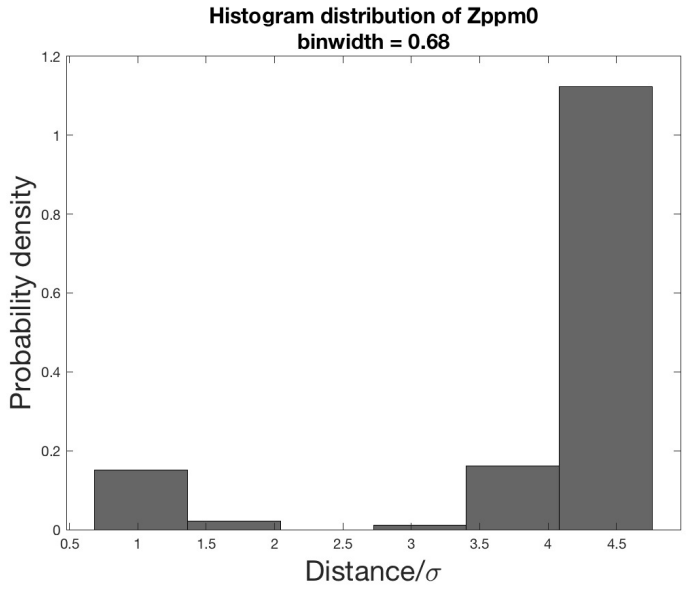


(a) without interleave

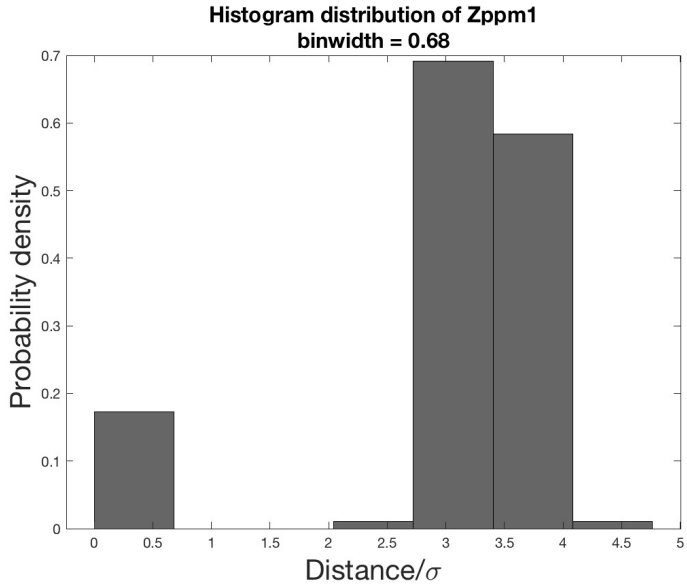


(b) with interleave

Figure 3.21: The 16 UNDHR text languages distances are computed by ppm and the distance matrix is shown by tree structure. Figure 3.21(a) shows the non-interleaving result and Figure 3.21(b) shows the interleaving result. The length of branches between the points correspond to the distances between languages.



(a) without interleave



(b) with interleave

Figure 3.22: The 16 UNDHR text languages distances are computed by ppm and the distance matrix are shown by histogram distributions. Figure 3.22(a) shows the non-interleaving result and the entropy value of the histogram is 2.52. Figure 3.22(b) shows the interleaving result and the entropy value of the histogram is 2.77.

3.3.4 Conclusion

Table 3.7: Entropy(top) and accuracy(bottom) values with histogram binwidth = 0.68.

	Zppm0	Zppm1	Zzip0	Zzip1	Zbzip0	Zbzip1
Entropy	2.52	2.77	2.49	2.77	2.5	2.54
Accuracy	1	1	1	1	1	1

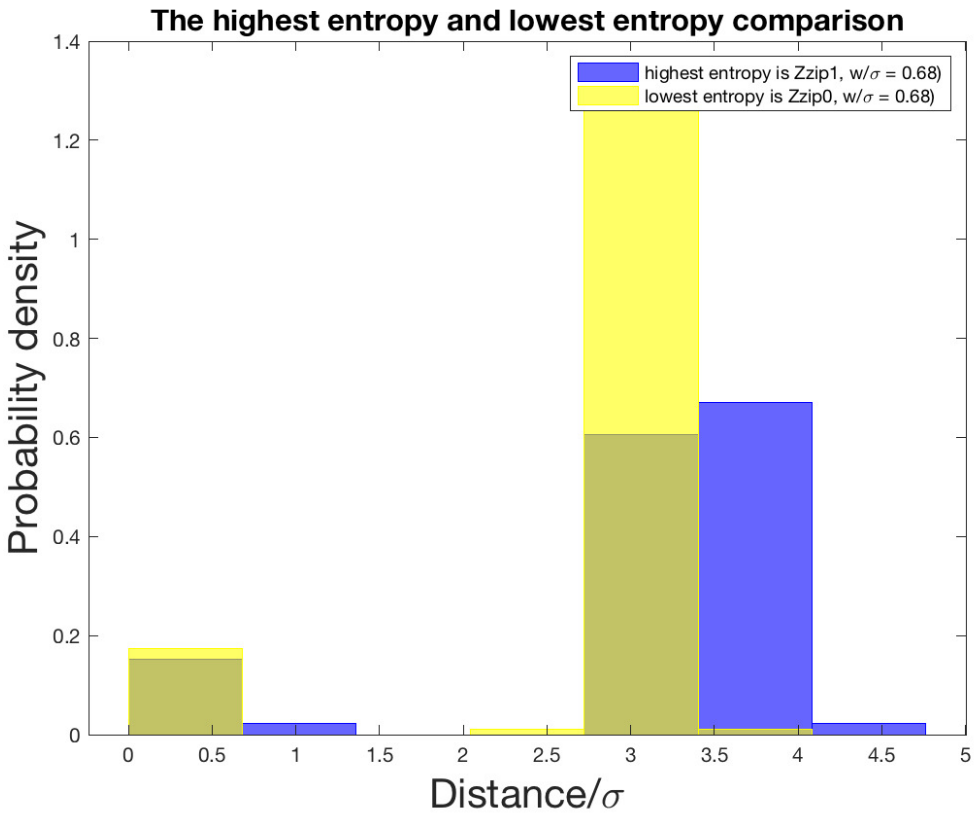


Figure 3.23: The histograms distributions of highest entropy and lowest entropy. The highest entropy is zip with interleaving and the lowest entropy is zip without interleaving.

In this section, we compare the language distances generated by 3 compressors: zip, bzip and PPM. The distance we used in the histogram is $Distance/\sigma$ and compared by probability density according to Section 3.2.1.2. Table 3.7 shows the entropy and accuracy for all zipping methods. The highest entropy is zip with interleaving and the lowest entropy is zip without interleaving. Figure 3.23 displays the histogram distributions of highest entropy (zip with interleave) and lowest entropy

(zip without interleave). By testing the probability of the null hypothesis of the highest and the lowest distance matrix, the p value of the t-test is $0.0017 < 0.01$ which rejects the null hypothesis H_0 that there is no difference between the means. So, we can say that the distribution of the highest entropy and the lowest entropy are significantly different. The bin width is w/σ, which is explained in Section 3.2.1.2 and the bin width is applied for zip, bzip and ppm with interleaving and non-interleaving results. This makes the histogram distribution too spiky to find the difference between the highest entropy and the lowest entropy. So we introduce the random distance matrix to calculate the random entropy and compare with those entropies. We run the random function for 1000 time with the same bin width and the average entropy of the random matrix is 1.0826. It shows that all entropy results are better than the random average result. However, the entropy here cannot fully describe the distance variations of languages since interleaving methods mix the character sets. Especially that when languages are very similar in this case, interleaving does not perform better grouping than non-interleave (as the word structure is destroyed by the interleaving method). And the languages trees also show that the non-interleaving results are more close to the linguistic language tree. Thus, to describe the language relationships by the highest entropy, the ppm without interleaving is the best choice for our task.

3.4 Conclusion

In this section, we have made an initial examination of the performance of TLID systems. We have looked at two classes of system. In the first class are those based on an approximation of the algorithm information theory distance ([Cilibrasi and Vitányi, 2005]). Instead of using the default penalty value, we compare the recognition results of different penalty cases. We study the n-gram distances by looking at the histogram distributions and entropies. We use the language distances to build the language tree which can be compared to the linguistic language trees. In the second class, the distances are built by zipping and build up the language trees by using the same method as n-gram method. It is well known that n-gram systems outperform zipping systems (indeed, this point was made strongly in a series of criticisms of zipping by Goodman [2002]). However, we do not merely want to solve the TLID problem (it is mostly solved anyway); we are interested in generating a meaningful distance between languages in the hope that we can use that distance to interpolate the missing distance in ALID and VLID. For this purpose, we need a method that can generate "good" distance matrices. In this chapter, we have considered what might constitute a "good" distance matrix and we have developed the concept of measuring the entropy of the distance matrix. As we previously mentioned, all n-gram results can be built up into language trees. Those language trees are compared with the linguistic language tree (shown in Figure 2.7 and Figure 2.8). Additionally, referring to the linguistic language tree can also avoid the random tree impact whose entropy might also be high. As we discussed in Section 3.3.3, the entropy describes the internal structure of the language distances. A high-entropy matrix provides more differentiation of distances than low-entropy matrices, which tend to be "all or nothing". For example, according to Figure 2.7, Catalan, Spanish and Portuguese are under the same sub-tree. So is it possible that Portuguese is closer to Spanish rather than Catalan? For linguistic language tree, it does not answer this question but our experiment tries to explain. So, the high entropy result might be more applicable to answer this question than low entropy.

This section, we have shown how, with a few parameters, both zipping methods and n-gram methods can produce "good" distance matrix. The next question, therefore, requires that we examine audio or video to see how we might map the domains to text. Before answering this question, however, it is wise to examine the performance of ALID specifically using some of our existing methods - is it viable to apply these text-based algorithms to audio? This is the topic of the next chapter.

Chapter 4

ALID (Audio Language IDentification) results

4.1 Introduction

In this chapter, we will introduce four techniques to compute audio language distances. Previously we have introduced two methods for TLID which performed with high accuracy. One is from Cavnar and Trenkle [1994]'s n-gram model and one is from Benedetto et al. [2002]'s zipping model based on relative entropy. Although the zipping methods do not show high entropy results, we wonder if these two techniques produce the same results as TLID. What we expect is, in ALID, the Cavnar and Trenkle [1994]'s n-gram model still gets a higher accuracy and a higher entropy than zipping methods. In ALID, we also examine Campana and Keogh [2010]'s CK-distance model based on MPEG compression by using the same database.

The database we used in this chapter is the Universal Declaration of Human Rights (UNDHR) dataset, which is a high-quality dataset downloaded from LibriVox[1]. A detailed description of the audio datasets was presented in Section 2.6.

[1]https://librivox.org/the-universal-declaration-of-human-rights-by-the-united-nations/

4.2 Feature extraction

The audio datasets contain recordings of the waveforms of utterances from each language signal collected from different speakers. For acoustic recognition and speech-to-text systems, the benchmark features are MFCCs.

4.2.1 MFCC (Mel-frequency cepstrum coefficient)

Mel-frequency cepstral coefficients (MFCCs) are an audio feature extraction technique, proposed by Rabiner and Juang [1993]. We use HTK which implemented by Cambridge University Engineering Department (CUED) for feature extraction and Figure 4.1 shows an overall MFCC feature extraction procedure.

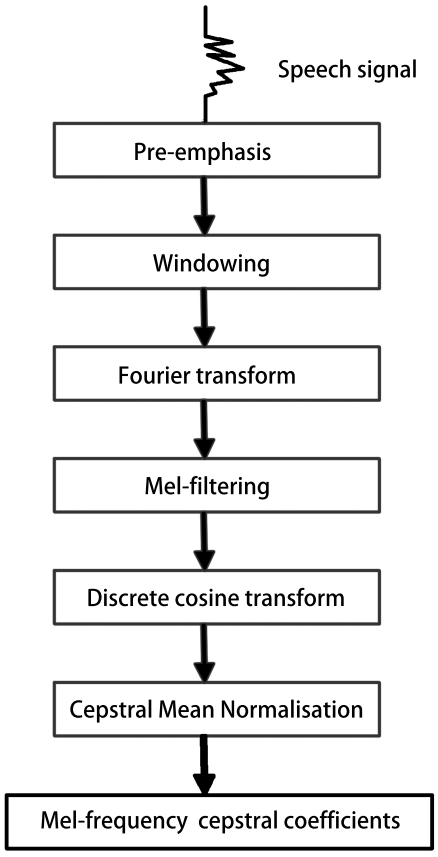


Figure 4.1: A standard example of MFCC feature extraction

The pre-emphasis stage removes the impact of the glottal pulses and radiation impedance [Markel and Gray, 2013], which is:

$$y(s_n) = s_n - ks_{n-1} \tag{4.1}$$

s means the symbol in a speech and $s \in 1...n$. k is a pre-defined parameter, for which $0 < k < 1$.

Since the Fourier transform is performed only on periodic samples, it is necessary to apply windowing techniques. In reality, the sample cannot be an integer number of periods, so the required windowing techniques should be able to reduce the boundary effect. [Young et al., 2006] use the Hamming window function (shown in 4.2).

$$y(s_n) = (0.54 - 0.46 \times \cos(\frac{2\pi(n-1)}{N-1})) \times s_n, n \in 1...N \tag{4.2}$$

The Discrete Fourier transform (DFT) converts a signal from the time domain to the frequency domain. After using the Hamming window function, the signal is framed into $10ms$ which allows overlap to the frames. For each frame, it calculates the frequency under the condition of n known samples s with a sample period T. The jth discrete time signal of Fourier coefficient c_j is equal to

$$c_j = 1/n \sum_{i=0}^{n-1} s_i \exp(-jik2\pi/n), 0 < |i| < N/2 \tag{4.3}$$

where k is the frequency and frequency$(s_n) = k \times T/n$ [Schilling and Harris, 2012].

Young et al. [2006] transform the FFT frequency sequence $\boldsymbol{f}$ to mel-scale $\boldsymbol{m}$, see equation 4.4. The mel-scale stage tries to imitate the human auditory system, which percepts signal frequency on a non-linear scale [Stevens et al., 1937]. It applies the triangular filter to each periodogram. The triangular filter can help to capture the spectrum energy with its shape.

$$\boldsymbol{m}_f = 2595 \times \log_{10}\left(1 + \frac{\boldsymbol{f}}{700}\right) \tag{4.4}$$

The discrete cosine transform (DCT) produces the MFCCs c_i by log filter bank amplitude, and the transmission equation is shown in equation 4.5 [Young et al., 2006]. **m** is the sequence that was produced by mel-scale previously.

$$c_i = \sqrt{\frac{2}{N}} \sum_{j=i}^{N} \boldsymbol{m}_j \cos(\frac{\pi i}{N}(j - 0.5)) \tag{4.5}$$

To remove long-term spectral effects, such as multiple microphones and room acoustics, Young et al. [2006] recommended cepstral mean normalisation, which can remove the cepstral mean of the transmission channels from all input vectors in the log cepstral domain.

The HTK parameters used for MFCC generation are summarised in Table 4.1

Table 4.1: MFCC parameters definition in HTK for audio files.

Parameters	**Value**	**Description**
SOURCEFORMAT	WAV	Definition of the format of the speech files. WAV stands for waveform.
TARGETKIND	$MFCC\_0\_D\_A$	Identifier of the coefficients to use. In this task, we used delta and acceleration coefficients with 0th cepstral coefficient.
WINDOWSIZE	250000.0 = 25 ms	Length of a time frame
TARGETRATE	100000.0 = 10 ms	Length of a frame period.
NUMCEPS	12	Number of MFCC coefficients
USEHAMMING	T	Use of Hamming function for windowing frames.
PREEMCOEF	0.97	Pre-emphasis coefficient.
NUMCHANS	26	Number of filterbank channels.
CEPLIFTER	22	Length of cepstral filtering.

Thus, using the parameters in Table 4.1, we converted each waveform into a matrix of HTK vectors of dimension 39 and a rate of $\frac{1}{10ms} = 100Hz$

4.3 Cavnar and Trenkle's N-gram model

In this section, we examine the Cavnar and Trenkle [1994]'s N-gram frequency model to discover the relationships between audio languages. We previously applied

the techniques of Cavnar and Trenkle [1994]'s N-gram frequency model in Section 3.2.1.1.

The dataset we used in ALID is the UNDHR dataset, which is provided by Librivox[2] and was previously described in Section 2.6.2. The UNDHR audio data were transformed into MFCCs using the procedure in Section 4.2.1.

4.3.1 Methods

Figure 4.2 is a modified TLID system but applied to ALID. A critical issue is how best to convert MFCCs to a text format for N-gram analysis. Our solution is to use a vector quantiser (VQ) system.

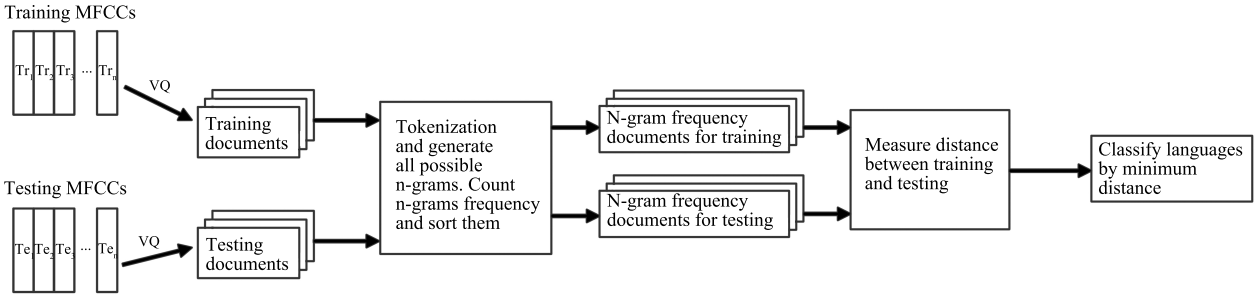


Figure 4.2: Cavnar and Trenkle [1994]'s n-gram frequency model for UNDHR audio dataset.

To use n-grams, it is necessary to convert MFCCs into discrete features. In that case, we use vector quantisation to divide MFCCs features into different bins. However, the index of the bins is not easy to make an n-gram list. A simple bigram case is that both 11 and 121, 111 and 21 can construct the bigram of 11121. To solve this problem, we define each bin is assigned to a unique Unicode character. So, the ALID n-gram frequencies can be calculated the same as the TLID. The detail of vector quantisation is in Section 4.3.1.1.

According to Figure 4.2, we first vector quantise the MFCCs into different bins. We examine 16, 32, 64, 128 and 256 bins to see which bin size gets the better per-

[2]https://librivox.org/the-universal-declaration-of-human-rights-by-the-united-nations/

formance. Each bin is represented by one character. The string of the characters is then analysed via Cavnar and Trenkle [1994]'s N-gram frequency model to calculate the frequency of each bin and hence calculate the language distances based on the bin frequency differences and averaged by the 10-fold cross validation. The detail of cross validation is in Section 3.2.1.3. We examine ALID by uni-gram, bi-gram, tri-gram, quad-gram and five-gram. Also as Cavnar and Trenkle [1994] mentioned, the maximum penalty of n-gram sequences is 400 without accepting all n-grams.

4.3.1.1 Vector quantisation

VQ (Vector quantisation) is one of the implementations of Shannon's sampling theorem for speech, image coding and compression from real vectors into digital representations. It is often considered to be a form of lossy data compression: outputting a digital signal from an analogue signal such as sound, temperature, light and pressure. VQ is widely applied in multiple areas such as communications, statistics and cluster analysis. Shannon implied that performance of coding vectors is always better than scalars [Gray, 1984]. Vector quantisation is useful for modelling symbolic data and reduce computation cost. In speech processing, vector quantisation creates a codebook and quantises each speech vector and give a unique symbol for each input frame. In this section, we use the HTK to create the codebook. Once a new MFCC feature comes in, vector quantisation compares the Mahalanobis distance $\boldsymbol{D}$ (4.6) between the features x and the means of the partitions μ in the codebook [Young et al., 2006]. Function 4.6 shows the Mahalanobis function which rescales the variables to make distances more comparable:

$$\boldsymbol{D} = \sqrt{(\boldsymbol{x} - \mu)^T \boldsymbol{S}^{-1} (\boldsymbol{x} - \mu)} \tag{4.6}$$

where $\boldsymbol{S}$ is the covariance matrix of $\boldsymbol{x}$ and μ and is calculated as $\boldsymbol{S} = \sqrt{\frac{\sum_{i=1}^{n}(x-\overline{x})(\mu-\overline{\mu})}{n-1}}$, which n is the length of $\boldsymbol{x}$, $\overline{x}$ is the mean of x and $\overline{\mu}$ is the mean of μ. In this section, we use the linear partition to create the codebook. The codebook saves the centroid of each bin. It firstly calculates the mean of the input features and divide features

by the mean value. Then, the mean is perturbed to generate two means and features are split based on which mean is nearest to them by using Equation 4.6. The means are then re-calculated by the split features and features are re-partitioned by the new means. These steps repeat until there is no significant total distortion. The total distortion is defined as the total distance between the features and the mean. Then the means are repeated the perturbing step until the required number of clusters are worked out[Young et al., 2006].

To determine how many clusters is sufficient for vector quantisation, we use the IPA (International Phonetic Alphabet) to calculate the number of phonemes exists in all languages (see Section 2.5.2). As the IPA contains 107 letters with 52 diaceitics and 4 prosodic marks, for vector quantisaiton, we assume that 256 clusters (bins) is sufficient to cover all phonemes corresponds to the total number of IPA characters. We start our experiment from the very small bin size, like 16 and 32, to see the impact of bin size variation. As we use 21 languages in our experiment, it is possible that not all phonemes in the IPA are used in these languages. So in that case, we also use the 64, 128 bins to see if a smaller binsize is enough to cover these phonemes in ALID instead of 256 bins. The vector quantised data are then applied to genereate the n-gram list.

4.3.2 Language distance results with Jake's data

Before we start to work on the UNDHR dataset, we firstly do an experiment on a small dataset which is collected from Jake's video data. As our UNDHR dataset only contains one speaker for each language, we need to measure the n-gram distance based on the multi-speakers. As we previously mentioned in Section 2.6, Jake's data has three languages which are English, Mandarin and Arabic. The audio waveforms are all converted into MFCCs and the MFCCs are all converted into symbols (Unicode characters). By using the n-gram model, the differences of n-grams frequencies can determine the differences between the languages. Considering the IPA (see Section 2.5.2), we use 64 VQ binsize and penalty of 100 with bigram

for this 3 languages experiment.

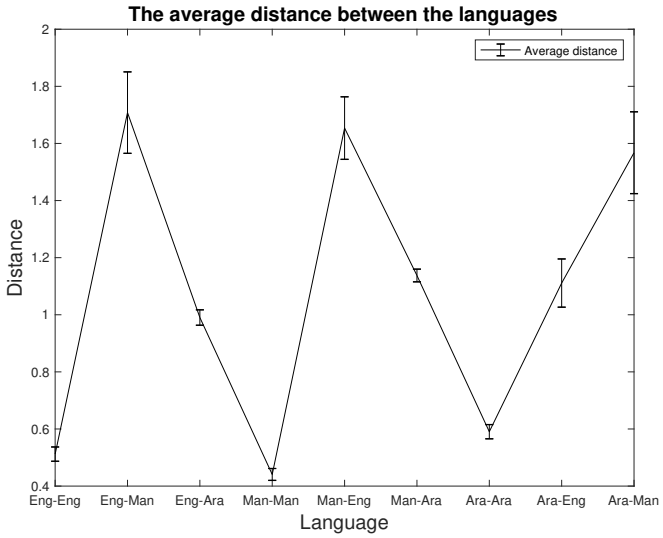


Figure 4.3: The n-gram distances between English, Mandarin and Arabic in ALID.

Figure 4.3 shows the average language distances between English, Mandarin and Arabic and the error bar on the average distance is the mean ± 2 standard error which obtains about 95% confidence interval of the estimate of the mean. We can find English, Mandarin and Arabic are all close to itself. The distances between the languages are far from the self-distance. As Jake's data has multi-speakers for each language with male and female speakers, we conclude that the n-gram distances represent inter-language differences rather than inter-speaker differences.

4.3.3 Language distance results with 16 bins

In this section, we examine the results of Cavnar and Trenkle [1994]'s model applied to the UNDHR dataset. As we previously explained in Section 4.3.1.1, the audio waveforms are all converted into MFCCs and the MFCCs are all vector quantised into indices which represented by corresponding symbols (Unicode characters). By using Cavnar and Trenkle [1994]'s n-gram model, the differences of n-grams frequencies can determine the differences between languages. We add a penalty to describe the impact of the n-grams which are not been seen in the other languages.

Tables 4.2 to 4.6 show the accuracy and entropy of each n-gram model with

different VQ bins. We use uni-grams through to five-grams. Both the accuracy and the entropy are measured using 10-fold cross validation. The accuracy and its standard error are computed as the mean and the standard error of the ten test accuracies from each folder, which using an n-gram classifier trained on the training data in each fold. Each fold also produces a distance matrix which are the distances between the test languages in that fold as measured by the n-gram method trained on each training fold. The mean of these distances is summarised by the entropy.

Table 4.2: Entrpoy values which binwidth = 0.57 vq bin size = 16.

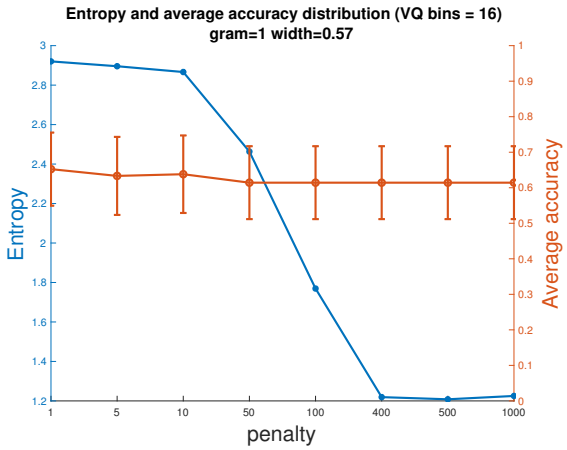
	Entropy							
Penalty value	1	5	10	50	100	400	500	1000
Gram=1	2.92	2.90	2.87	2.46	1.77	1.22	1.21	1.23
Gram=2	2.89	2.88	2.89	2.87	2.86	2.62	2.67	2.49
Gram=3	2.80	2.75	2.74	2.72	2.72	2.75	2.77	2.81
Gram=4	2.72	2.71	2.63	2.12	2.27	2.63	2.61	2.69
Gram=5	2.77	2.77	2.78	2.48	2.51	2.83	2.86	2.85
	Accuracy value							
Gram=1	0.65	0.63	0.64	0.61	0.61	0.61	0.61	0.61
Gram=2	0.43	0.48	0.54	0.76	0.57	0.28	0.26	0.16
Gram=3	0.20	0.20	0.21	0.57	0.67	0.57	0.55	0.50
Gram=4	0.07	0.08	0.10	0.47	0.65	0.61	0.60	0.59
Gram=5	0.03	0.03	0.03	0.50	0.70	0.67	0.68	0.61
	Standard error							
Gram=1	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
Gram=2	0.06	0.06	0.06	0.05	0.05	0.04	0.04	0.03
Gram=3	0.04	0.04	0.04	0.05	0.05	0.05	0.06	0.05
Gram=4	0.02	0.02	0.03	0.04	0.04	0.05	0.05	0.06
Gram=5	0.02	0.02	0.02	0.04	0.04	0.04	0.04	0.04

Table 4.2 shows the accuracy and entropy of Cavnar and Trenkle [1994]'s n-gram model with 16 VQ bins. Figure 4.4 compares the accuracies and entropies, the accuracy has error bars with mean ±2 standard error. We find the highest accuracy is the bi-gram (Figure 4.4(b)), whose penalty is 50. Like we previously explained in Section 4.3.1.1, we use the Unicode characters to represent the index of VQ bins. For example, in 16 VQ bins with unigram case, the Unicode characters which are transformed from the index of VQ bins which contains the Arabic phonemes are "NOIGMDLHFCJEPKBA_" and the English one is "ABCDGEFMKLHNJIO_".

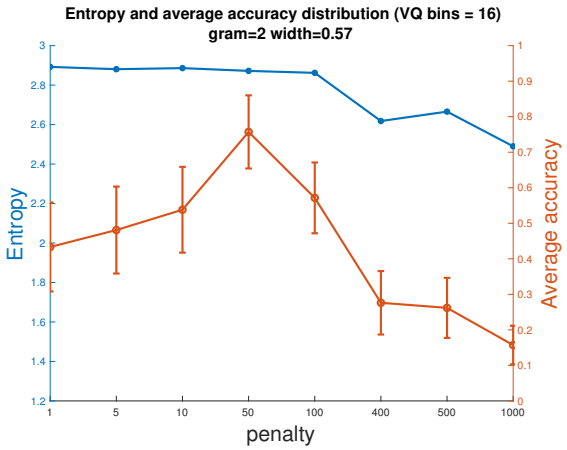
We can find these two string contains the same characters, so these two languages share the same character set. For uni-gram with 16 VQ bins, there are only 16 n-grams and most of the languages have the same character set except Czech, English, Russian and Vietnamese. So the accuracy is similar for all penalties. Considering the entropy (blue lines) in Figure 4.4(e), we see two effects. For the low-order n-grams (uni-grams and bi-grams), the major effect of the penalty is to add language distances onto language pairwise distances if they do not share an n-gram. This makes the distance distribution spikier and lowers entropy. As we move to high-order n-grams, the list of comparable n-gram grows but is capped at 400. This cap has the effect of flattening the distance distribution since the language pairwise distance involves distance n-grams that no longer appear on the list. Thus we see two effects with the increasing penalty - decreasing and increasing entropy leading to a characteristic dip in entropy in the mid-penalty region for longer n-grams. We want high accuracy and, for later work, we shall want high entropy, which here imply bi-grams with a penalty of 50.

Figure 4.5 visualizes the bi-gram, 50 penalty result in 16 VQ bins. Figure 4.5(a) shows the colour map of languages and Figure 4.5(b) shows the dendrogram which is built based on $d = distance/\sigma$ where d is normalized into $[0, 1]$. The dendrogram is built based on complete-linkage clustering (explained in Section 3.2.2). According to the linguistic language tree in Section 2.7, we can define three language subsets - Spanish and Portuguese, Korean and Japanese, Czech and Polish. In the colour maps, we denote the Spanish and Portuguese in pink, Korean and Japanese in blue colour and Czech and Polish in red colour. In the dendrogram, we denote Spanish and Portuguese as "$", Korean and Japanese as symbol "$*$" and Czech and Polish as symbol "+". We find Portuguese is close to Polish, Spanish and Swedish. Czech is also close to Polish, Spanish and Swedish. It is possible that they show low distances between each other because they are all Indo-Hittite languages. The distance between the Japanese and Korean is far from each other but is also positive that the distances between Japanese and English and Russian are longer than the Japanese and Korean. Thus, we can conclude that the language tree which is built

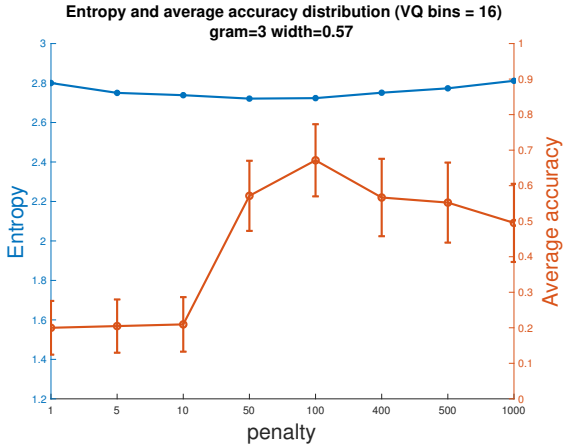
by the 16 bins might not previously describe the language relationships. So we are going to study the 32 bins case and see if the language relationships can be presented better alongside the VQ binsize.



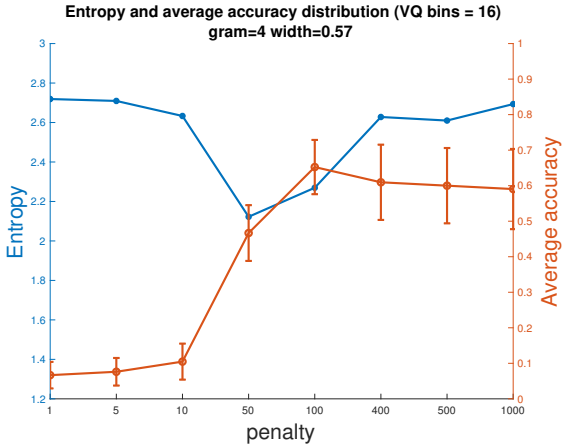
(a) Accuracy and entropy distribution for gram = 1



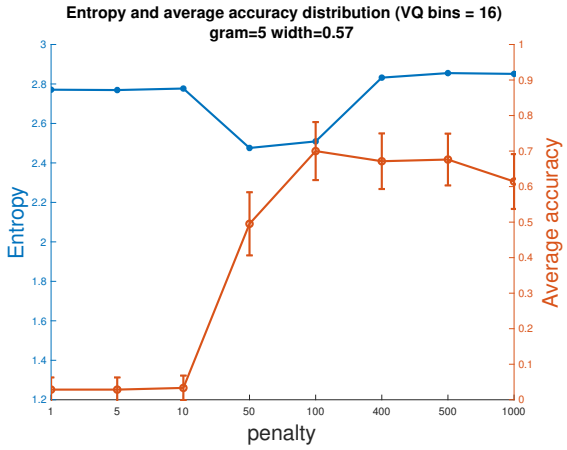
(b) Accuracy and entropy distribution for gram = 2



(c) Accuracy and entropy distribution for gram = 3

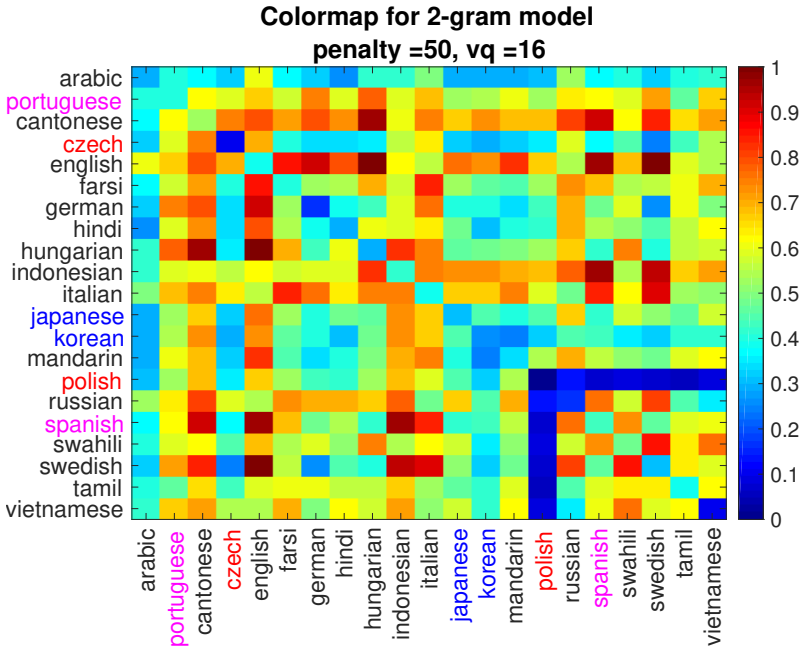


(d) Accuracy and entropy distribution for gram = 4

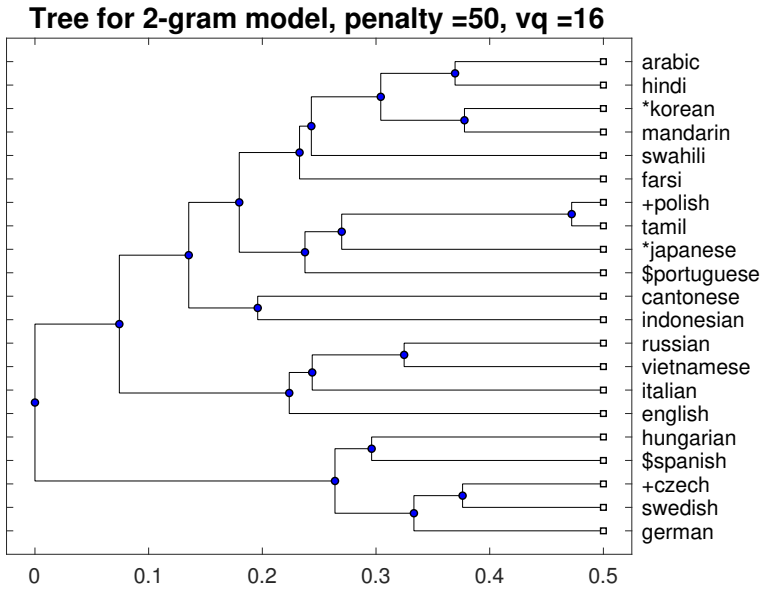


(e) Accuracy and entropy distribution for gram = 5

Figure 4.4: Accuracy and entropy distribution for n-grams. VQ bin size is 16. The x-axis is the penalty value. The left y-axis is the entropy value and the right y-axis is the accuracy value. The error bar on the average accuracy is the mean ± 2 standard error which obtains about 95% confidence interval of the estimate of the mean.



(a) Colour map of bi-gram



(b) Dendrogram of bi-gram

Figure 4.5: The 21 UNDHR audio language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 50 and the VQ bins is 16. Figure 4.5(a) shows the colour map of the language distance variations and Figure 4.5(b) shows the language tree which is built by the distances. The colour variation in Figure 4.5(a) shows the pairwise distances between languages.

4.3.4 Language distance results with 32 bins

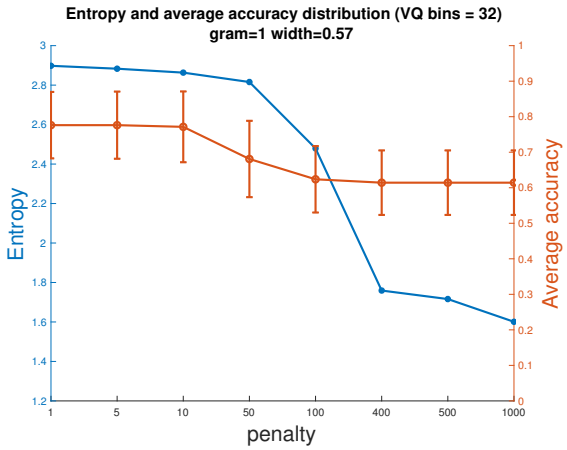
Table 4.3: Entrpoy values which binwidth = 0.57 vq bin size = 32.

	Entropy							
Penalty value	1	5	10	50	100	400	500	1000
Gram=1	2.90	2.88	2.86	2.82	2.48	1.76	1.72	1.60
Gram=2	2.86	2.78	2.76	2.78	2.78	2.85	2.88	2.85
Gram=3	2.67	2.65	2.57	2.09	1.96	2.55	2.60	2.63
Gram=4	2.74	2.74	2.65	1.96	2.05	2.64	2.69	2.65
Gram=5	2.79	2.77	2.77	2.52	2.48	2.83	2.89	2.88
	Accuracy value							
Gram=1	0.78	0.78	0.77	0.68	0.62	0.61	0.61	0.61
Gram=2	0.21	0.21	0.22	0.39	0.86	0.51	0.45	0.40
Gram=3	0.07	0.07	0.08	0.33	0.72	0.68	0.66	0.65
Gram=4	0.04	0.05	0.06	0.30	0.75	0.74	0.73	0.71
Gram=5	0.02	0.03	0.04	0.30	0.76	0.76	0.75	0.75
	Standard error							
Gram=1	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
Gram=2	0.04	0.04	0.04	0.06	0.03	0.06	0.05	0.05
Gram=3	0.03	0.03	0.03	0.04	0.05	0.05	0.05	0.05
Gram=4	0.01	0.02	0.02	0.04	0.04	0.04	0.04	0.05
Gram=5	0.01	0.01	0.01	0.03	0.04	0.04	0.04	0.04

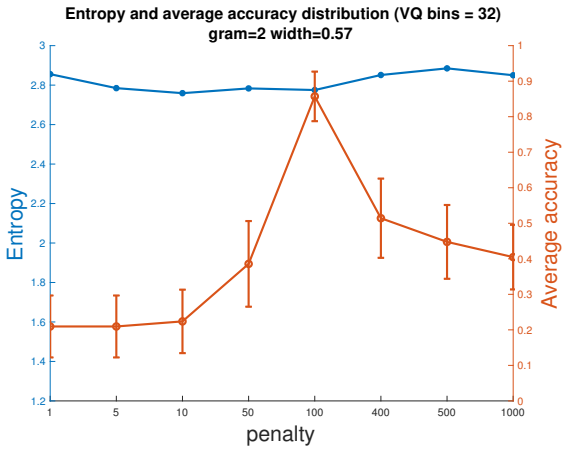
Table 4.3 shows the accuracy and entropy of Cavnar and Trenkle [1994]'s n-gram model with 32 VQ bins. Figure 4.6 compares the accuracies and entropies and comparable to Figure 4.4. We find the accuracy and entropy distribution of 32 VQ bins are similar to the 16 bins. The highest accuracy in 32 VQ bins is with bi-gram and the penalty is 100 (rather than 50 in 16 VQ bins). As the number of VQ bins increases up to 32 (up to 1056 in bi-gram but the model only accept the highest rank of 400 n-grams), the variation of language character set shows more differentiation of languages, which means the model needs a higher penalty to identify languages. We can see it is worth to use higher penalty values since we get a higher accuracy rather than the results of 16 VQ bins. We can conclude the best performance in 32 VQ bins with bi-grams with 100 penalty.

Figure 4.7 shows the same diagram as 4.5 but with bi-grams, 100 penalty result in 32 VQ bins. Figure 4.5(a) shows the colour map of languages and Figure 4.5(b) shows

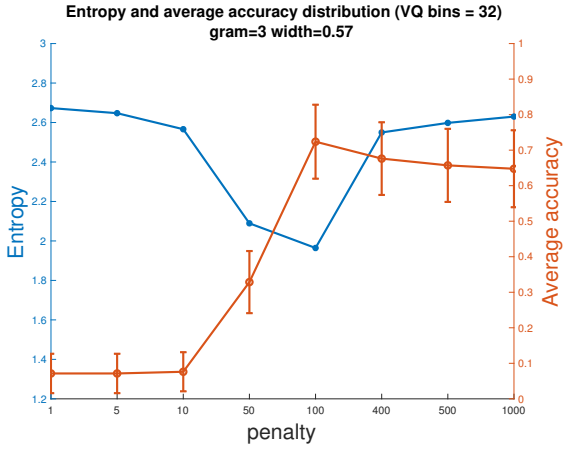
the complete-linkage clustering dendrogram which is built based on $d = distance/\sigma$ and the d is normalized into $[0, 1]$. We can find most of language distances are far from each other as the penalty is 100. Although through the language tree, we find the distances between Czech and Polish, Spanish and Portuguese, Japanese and Korean are not the closest, we still can find the distance between Indo-Hittite languages are closer than other languages. The Japanese and Korean language is also represented poorly as the 16 bins case. The colour map shows that most of distances are larger than Figure 4.5, which is because the penalty is 100 instead of 50 and contains more "out-of-place" n-grams.



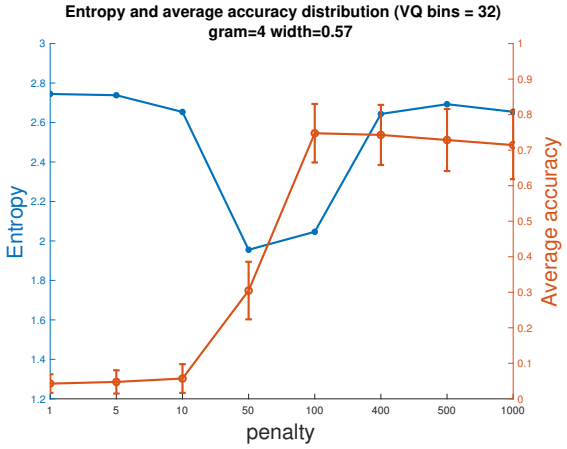
(a) Accuracy and entropy distribution for gram = 1



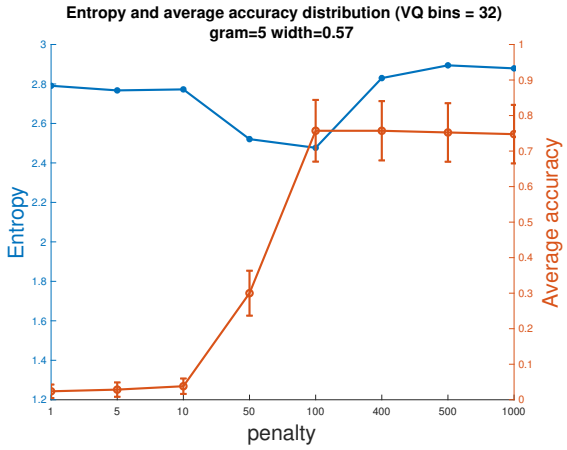
(b) Accuracy and entropy distribution for gram = 2



(c) Accuracy and entropy distribution for gram = 3

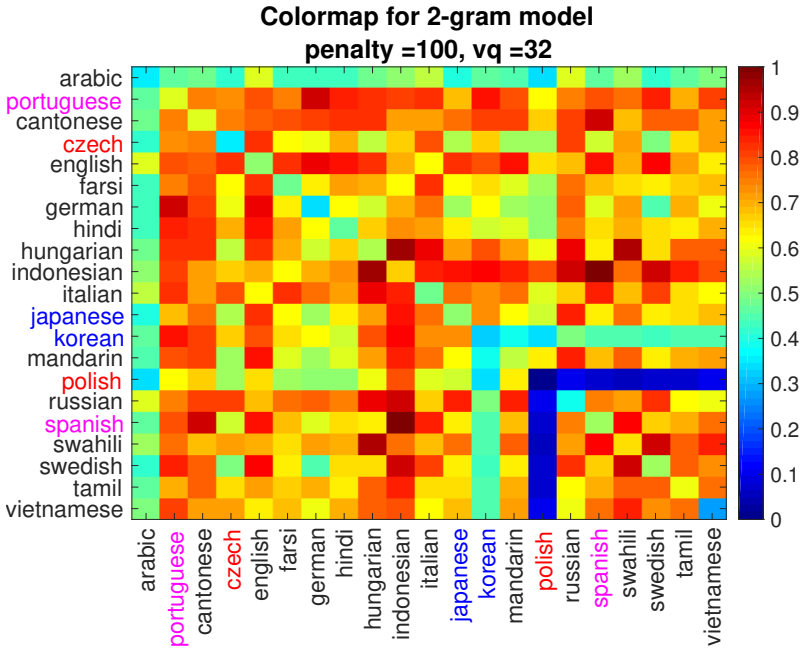


(d) Accuracy and entropy distribution for gram = 4

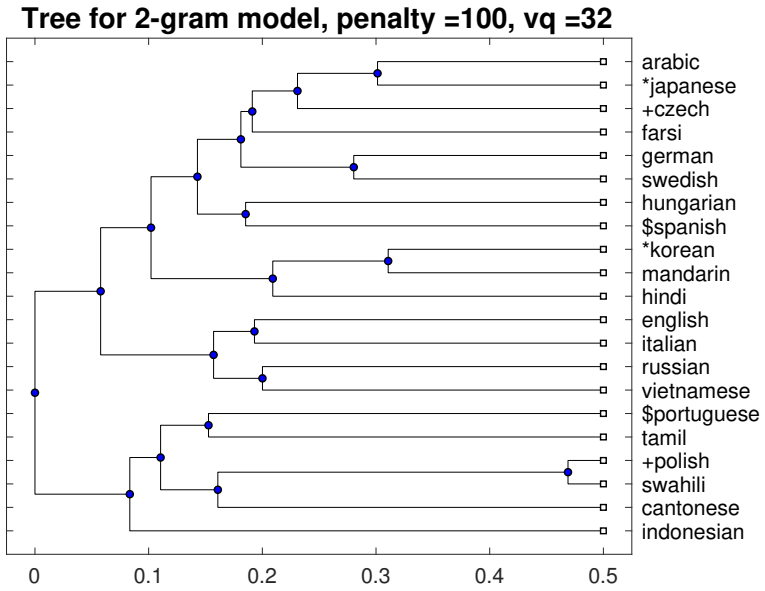


(e) Accuracy and entropy distribution for gram = 5

Figure 4.6: Accuracy and entropy distribution for n-grams. VQ bin size is 32. The x-axis is the penalty value. The left y-axis is the entropy value and the right y-axis is the accuracy value. The error bar on the average accuracy is the mean ± 2 standard error which obtains about 95% confidence interval of the estimate of the mean.



(a) Colour map of bi-gram



(b) Dendrogram of bi-gram

Figure 4.7: The 21 UNDHR audio language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 100 and the VQ bins is 32. Figure 4.7(a) shows the colour map of the language distance variations and Figure 4.7(b) shows the language tree which is built by the distances. The colour variation in Figure 4.7(a) shows the pairwise distances between languages.

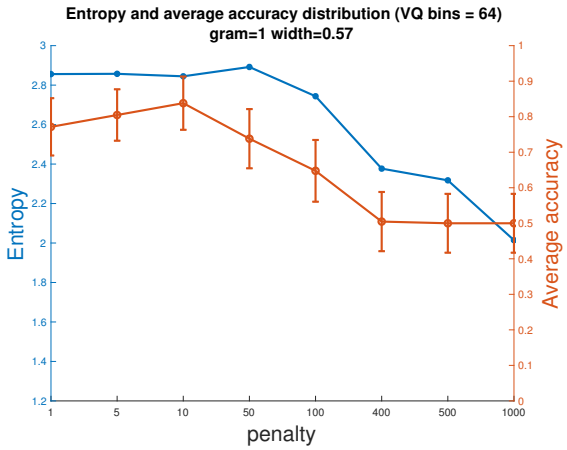
4.3.5 Language distance results with 64 bins

Table 4.4: Entrpoy values which binwidth = 0.57 vq bin size = 64.

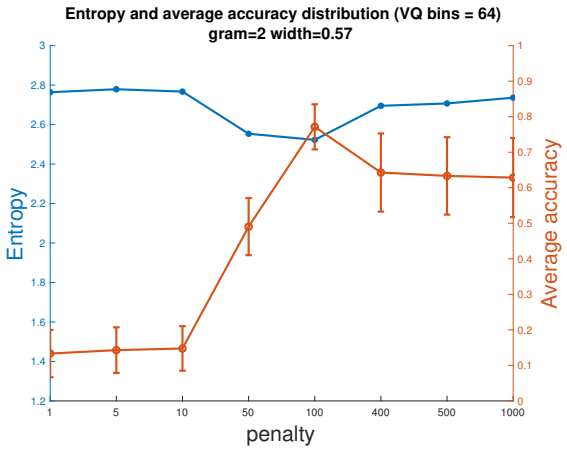
	Entropy							
Penalty value	1	5	10	50	100	400	500	1000
Gram=1	2.86	2.86	2.84	2.89	2.74	2.38	2.32	2.01
Gram=2	2.76	2.78	2.77	2.55	2.52	2.70	2.71	2.74
Gram=3	2.73	2.69	2.63	1.81	1.83	2.30	2.34	2.44
Gram=4	2.87	2.80	2.81	2.02	2.06	2.57	2.56	2.66
Gram=5	2.86	2.87	2.83	2.45	2.50	2.85	2.89	2.86
	Accuracy value							
Gram=1	0.77	0.80	0.84	0.74	0.65	0.50	0.50	0.50
Gram=2	0.13	0.14	0.15	0.49	0.77	0.64	0.63	0.63
Gram=3	0.03	0.05	0.07	0.25	0.80	0.74	0.71	0.70
Gram=4	0.02	0.03	0.03	0.28	0.81	0.78	0.77	0.75
Gram=5	0.03	0.05	0.05	0.35	0.84	0.80	0.80	0.78
	Standard error							
Gram=1	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04
Gram=2	0.03	0.03	0.03	0.04	0.03	0.06	0.05	0.06
Gram=3	0.02	0.02	0.02	0.05	0.04	0.05	0.05	0.05
Gram=4	0.01	0.02	0.02	0.05	0.05	0.04	0.03	0.04
Gram=5	0.01	0.01	0.01	0.04	0.03	0.04	0.04	0.04

Table 4.4 shows the accuracy and entropy of Cavnar and Trenkle [1994]'s n-gram model with 64 VQ bins. Figure 4.8 compares the accuracies and entropies with the accuracy is augmented by error bars. As we previously explained in Section 4.3.1.1, we use the Unicode characters to represent the index of VQ bins. Comparing with 16 bins case, as the VQ bins increase to 64, the uni-gram accuracy varies because most of the languages do not share the same bins, in another word, the character set (only four languages do not share the same character set in 16 VQ bins). The highest accuracy in the 64 VQ bins is uni-gram and the penalty is 10 rather than the bi-gram in the 32 VQ bins. And also, the variation of language character set shows more differences between languages. In this case, we conclude that the VQ bins do impact on the accuracy and entropy, which is not surprising that uni-gram gets a high accuracy. Thus, in the 64 bins case, we summarise the best performance is the uni-gram with 10 penalty.

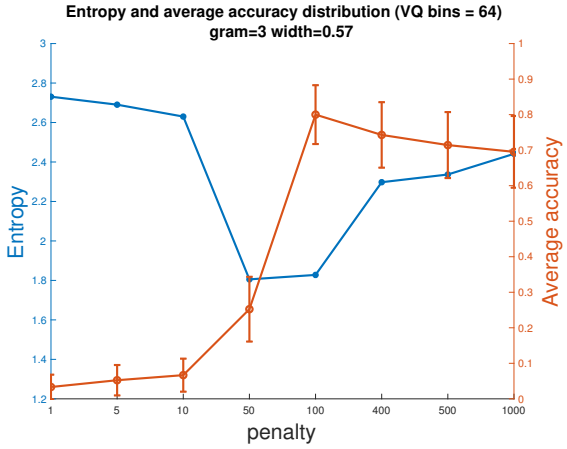
Figure 4.9(a) shows the same diagram as 4.5 with uni-gram, 10 penalty result in 64 VQ bins. Figure 4.9(a) shows the colour map of languages and Figure 4.9(b) shows the complete-linkage clustering dendrogram which is built based on $d = distance/\sigma$ and the d is normalized into $[0, 1]$. We can see 10 penalty shows more distance variation than 32 bins with 100 penalty. Although thorough the language tree, we find the distances between Czech and Polish, Spanish and Portuguese, Japanese and Korean are not the closest, we still find the distance between Indo-Hittite languages are closer than other languages.



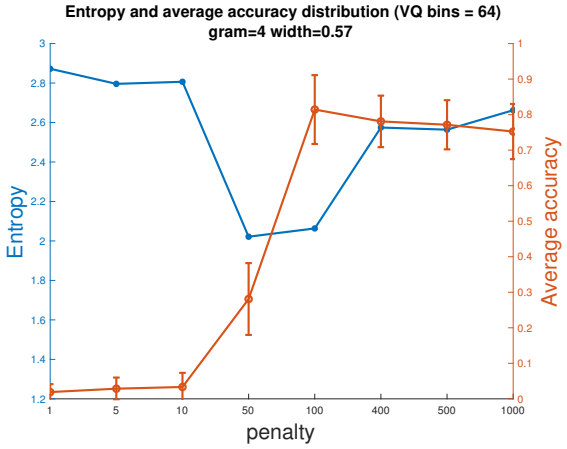
(a) Accuracy and entropy distribution for gram = 1



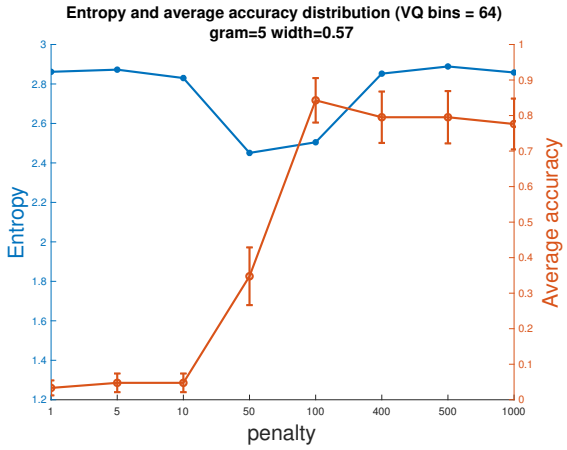
(b) Accuracy and entropy distribution for gram = 2



(c) Accuracy and entropy distribution for gram = 3

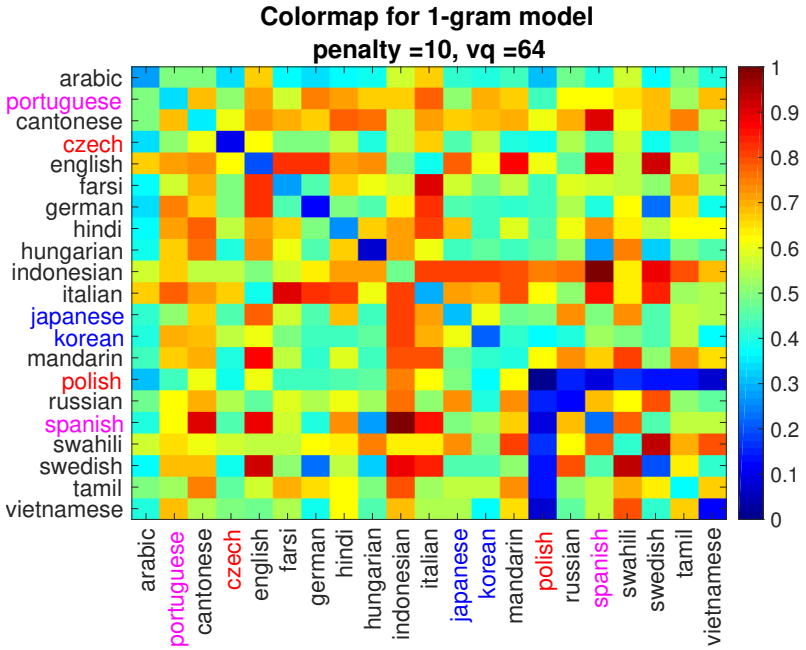


(d) Accuracy and entropy distribution for gram = 4

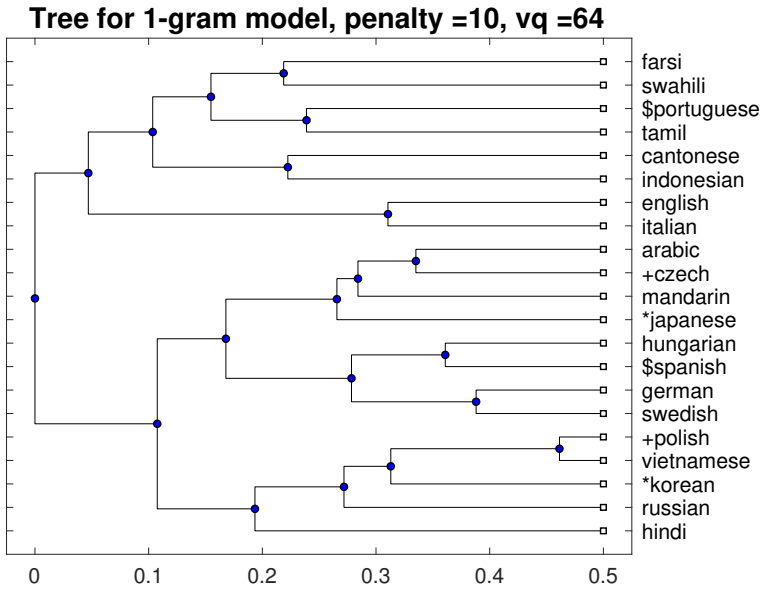


(e) Accuracy and entropy distribution for gram = 5

Figure 4.8: Accuracy and entropy distribution for n-grams. VQ bin size is 64. The x-axis is the penalty value. The left y-axis is the entropy value and the right y-axis is the accuracy value. The error bar on the average accuracy is the mean ± 2 standard error which obtains about 95% confidence interval of the estimate of the mean.



(a) Colour map of uni-gram



(b) Dendrogram of uni-gram

Figure 4.9: The 21 UNDHR audio language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 10 and the VQ bins is 64. Figure 4.9(a) shows the colour map of the language distance variations and Figure 4.9(b) shows the language tree which is built by the distances. The colour variation in Figure 4.9(a) shows the pairwise distances between languages.

4.3.6 Language distance results with 128 bins

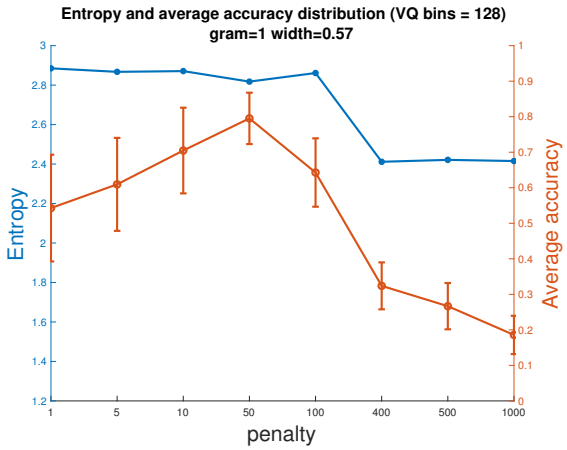
Table 4.5: Entrpoy values which binwidth = 0.57 vq bin size = 128.

	Entropy							
Penalty value	1	5	10	50	100	400	500	1000
Gram=1	2.88	2.87	2.87	2.82	2.86	2.41	2.42	2.42
Gram=2	2.65	2.67	2.62	2.21	2.11	2.51	2.51	2.55
Gram=3	2.73	2.67	2.61	1.77	1.51	2.25	2.25	2.38
Gram=4	2.84	2.80	2.73	2.03	2.12	2.60	2.59	2.72
Gram=5	2.86	2.85	2.80	2.39	2.52	2.88	2.87	2.88
	Accuracy value							
Gram=1	0.54	0.61	0.70	0.80	0.64	0.32	0.27	0.19
Gram=2	0.04	0.05	0.05	0.24	0.71	0.68	0.68	0.66
Gram=3	0.03	0.03	0.04	0.27	0.79	0.73	0.73	0.71
Gram=4	0.05	0.05	0.06	0.33	0.84	0.80	0.79	0.77
Gram=5	0.05	0.07	0.08	0.39	0.85	0.80	0.80	0.80
	Standard error							
Gram=1	0.08	0.07	0.06	0.04	0.05	0.03	0.03	0.03
Gram=2	0.02	0.03	0.03	0.03	0.04	0.05	0.05	0.05
Gram=3	0.02	0.02	0.02	0.04	0.04	0.04	0.04	0.05
Gram=4	0.01	0.01	0.01	0.04	0.04	0.03	0.03	0.04
Gram=5	0.01	0.01	0.01	0.04	0.03	0.03	0.03	0.03

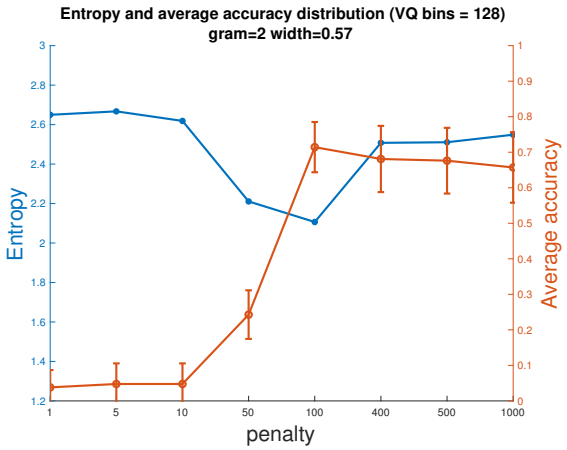
Table 4.5 shows the accuracy and entropy of Cavnar and Trenkle [1994]'s n-gram model with 128 VQ bins. Figure 4.10 compares the accuracies and entropies, the accuracy has error bars with mean ±2 standard error which gives an approximate 95% confidence interval. The distributions of accuracy and entropy are similar to 64 VQ bins. The penalty value has a greater impact on uni-gram as the increasing of VQ bins. Also, as the VQ bin increase to 128, the distributions of accuracy and entropy became similar between tri-gram, quad-gram and five-gram. It tells us as the increasing of VQ bins, the impact of n-gram variation is less important compared to the penalty and VQ bins. In the 128 VQ bin results, the uni-gram with 50 penalty value is as the same accuracy as the five-gram with 400, 500 and 1000 penalty values. Considering the entropy value, we conclude that the best performance in 128 is the five-gram with 400 penalty.

Figure 4.11 shows the same diagram as 4.5 with five-gram, 400 penalty result

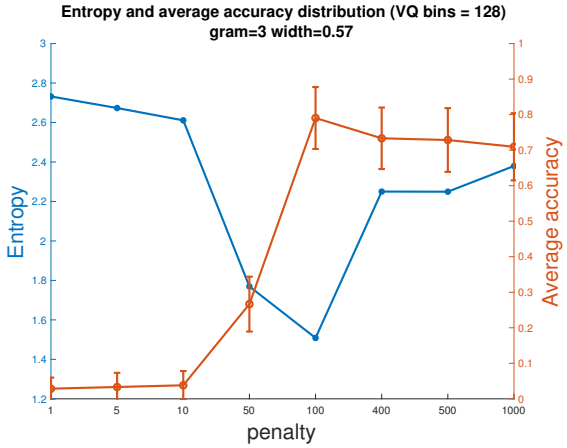
in 128 VQ bins. Figure 4.11(a) shows the colour map of languages and Figure 4.11(b) shows the complete-linkage clustering dendrogram which is built based on $d = distance/\sigma$ and the d is normalized into $[0, 1]$. We can find as the penalty grows up to 400, more language distances become unrelated. In 128 bins, we found the linguistically closed languages are still not close to each other. However, we find Polish is still always close to the part of Indo-Hittite languages like Russian, Spanish, Swahili and Swedish. It shows that the language distance structure is not highly impacted by the VQ bins if the number of bins contains sufficient information about the speech. Thus, we can say that the VQ binsize impact the accuracy of the n-gram language identification. Additionally, although Swahili is not an Indo-European language, it shares phonetic rules like nasal assimilation (For example, "good morning", the "d" in "good" is dropped in a rapid speech in English) with Spanish and other languages. However, Kučera and Monroe [1968] mentioned that the rules which wildly occurs are useless in linguistic language classification, which means the language grouping is based on unique rules in the linguistic area. According to Figure 2.7, there is no other Niger-Kordofanian language except Swahili in the dataset. So, it is not surprising that Swahili is close to Russian and Polish as they share part of phonemes and rules. And for the same reason, although Tamil is Dravidian language, Swahili is Niger-Congo language and Vietnamese is Austric language, they might share some common phoneme rules with Polish which are high occurrences but not considered by linguists for language grouping (but the model use them as the rules for language relationships).



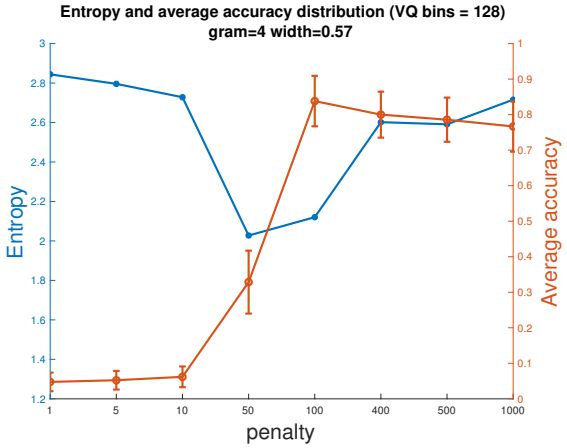
(a) Accuracy and entropy distribution for gram = 1



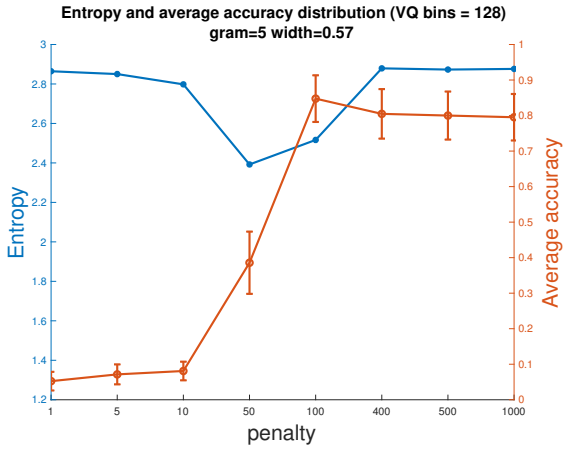
(b) Accuracy and entropy distribution for gram = 2



(c) Accuracy and entropy distribution for gram = 3

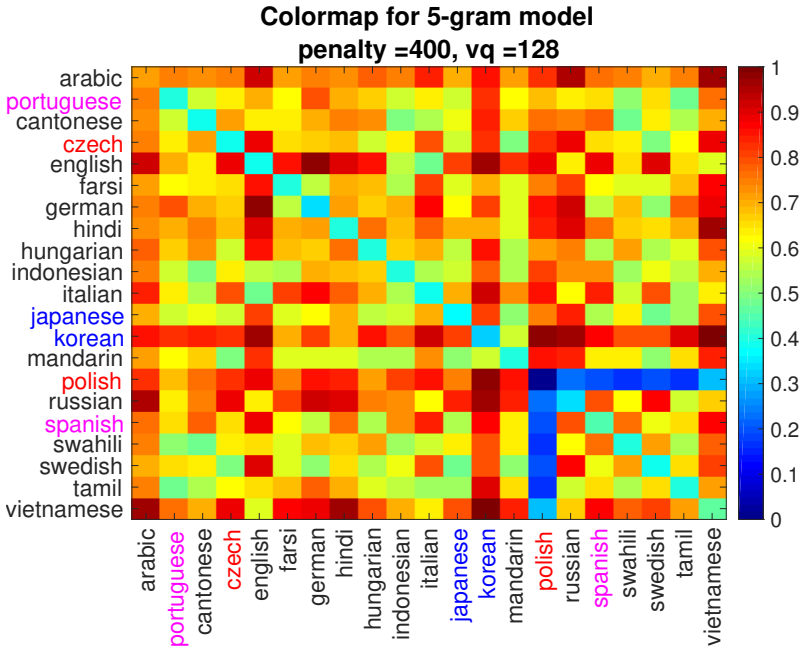


(d) Accuracy and entropy distribution for gram = 4

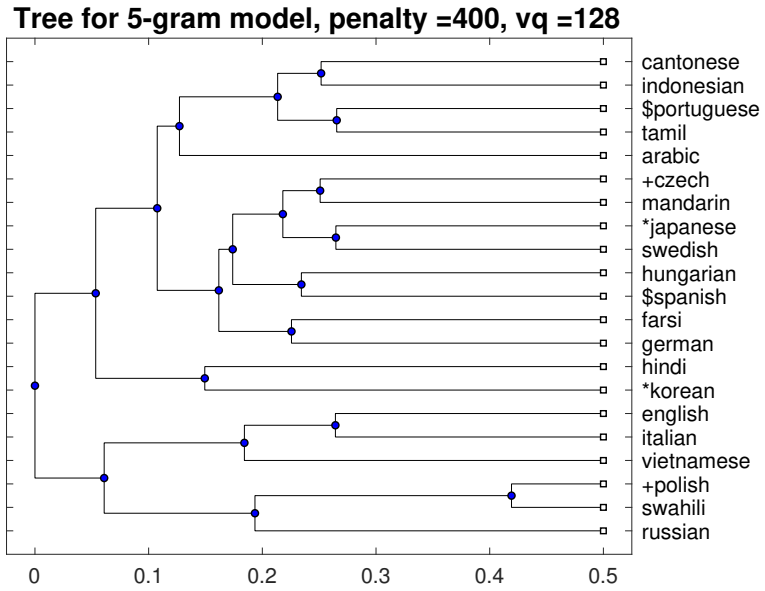


(e) Accuracy and entropy distribution for gram = 5

Figure 4.10: Accuracy and entropy distribution for n-grams. VQ bin size is 128 The x-axis is the penalty value. The left y-axis is the entropy value and the right y-axis is the accuracy value. The error bar on the average accuracy is the mean ± 2 standard error which obtains about 95% confidence interval of the estimate of the mean.



(a) Colour map of five-gram



(b) Dendrogram of five-gram

Figure 4.11: The 21 UNDHR audio language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 400 and the VQ bins is 128. Figure 4.11(a) shows the colour map of the language distance variations and Figure 4.11(b) shows the language tree which is built by the distances. The colour variation in Figure 4.11(a) shows the pairwise distances between languages.

4.3.7 Language distance results with 256 bins

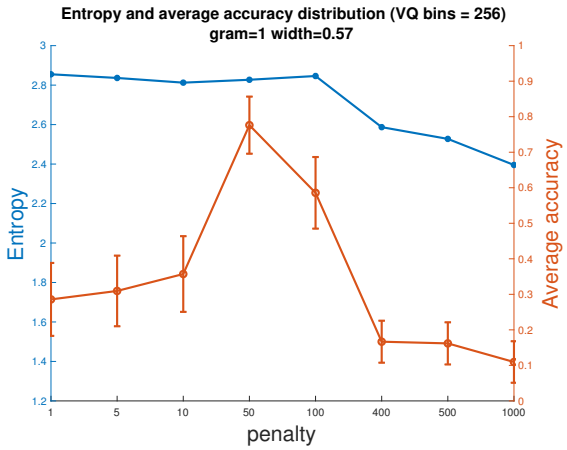
Table 4.6: Entrpoy values which binwidth = 0.57 vq bin size = 256.

	Entropy							
Penalty value	1	5	10	50	100	400	500	1000
Gram=1	2.85	2.84	2.81	2.83	2.85	2.59	2.53	2.40
Gram=2	2.81	2.77	2.69	2.42	2.56	2.71	2.73	2.77
Gram=3	2.81	2.72	2.59	1.92	1.78	2.32	2.38	2.45
Gram=4	2.78	2.74	2.69	2.09	1.98	2.35	2.37	2.42
Gram=5	2.80	2.74	2.70	2.22	2.07	2.47	2.48	2.53
	Accuracy value							
Gram=1	0.29	0.31	0.36	0.78	0.59	0.17	0.16	0.11
Gram=2	0.11	0.12	0.13	0.20	0.57	0.57	0.58	0.56
Gram=3	0.08	0.08	0.09	0.21	0.61	0.60	0.59	0.54
Gram=4	0.11	0.12	0.14	0.33	0.65	0.61	0.60	0.58
Gram=5	0.10	0.11	0.13	0.36	0.67	0.61	0.60	0.57
	Standard error							
Gram=1	0.05	0.05	0.05	0.04	0.05	0.03	0.03	0.03
Gram=2	0.03	0.04	0.04	0.04	0.05	0.05	0.05	0.06
Gram=3	0.02	0.02	0.02	0.03	0.05	0.06	0.06	0.06
Gram=4	0.03	0.03	0.04	0.03	0.05	0.06	0.05	0.05
Gram=5	0.02	0.03	0.03	0.03	0.05	0.06	0.05	0.05

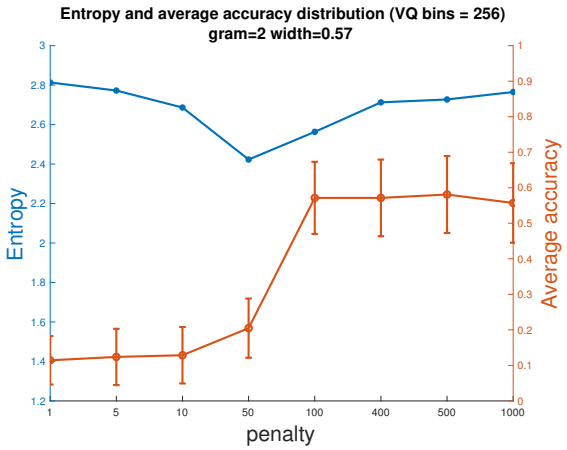
Table 4.6 shows the accuracy and entropy of Cavnar and Trenkle [1994]'s n-gram model with 256 VQ bins. Figure 4.12 compares the accuracies and entropies, the accuracy has error bars with mean ±2 standard error. The distributions of accuracy and entropy are similar to the 128 VQ bins. The 256 VQ bins results provide evidence that the penalty value has a greater impact on uni-gram as the increasing of VQ bins. Also, as the VQ bin increase to 256, the distributions of accuracy and entropy are similar between tri-gram, quad-gram and five-gram, which also proves that as the increasing of the number of VQ bins, the impact of n-gram variation is less important compared to the penalty and VQ bins. However, the accuracy of the 256 VQ bins is lower than the 128 VQ bins. It is because a large number of VQ bins lose more information during transforming from MFCCs to Unicode characters and can make the recognition accuracy worse. For example, either the VQ binsize is too large or small, the n-gram might view the different phonemes as the same or view the similar phonemes as different. In small binsize case, this might wrongly cause

the long repeat strings in small binsize case. In large binsize case, making similar phonemes different might change the rank of n-gram occurrences and decreasing the accuracy. In the 256 VQ bin results, the unigram with 50 penalty value shows the highest accuracy and the entropy is also relatively high. In this case, we conclude that the best performance in 256 is the uni-gram with 50 penalty value.

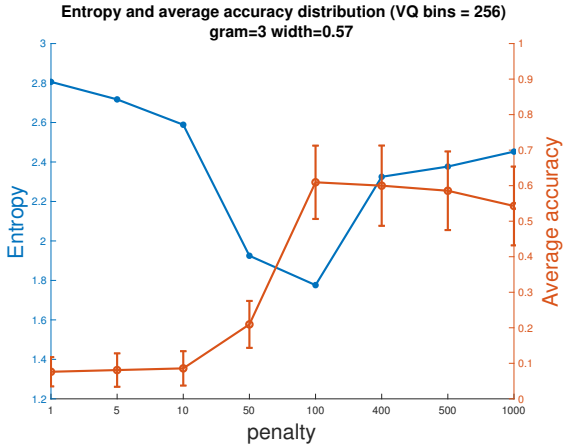
Figure 4.13 shows the same diagram as 4.5 with uni-gram, 50 penalty result in 256 VQ bins. Figure 4.13(a) shows the colour map of languages and Figure 4.13(b) shows the complete-linkage clustering dendrogram which is built based on $d = distance/\sigma$ and the d is normalized into $[0, 1]$. In 256 bins, we found the linguistically closed languages are still not close to each other. Figure 4.13 shows that Polish is still close to part of the Indo-Hittite languages like Russian and Swahili while the distance variations are smaller than 64 bins.



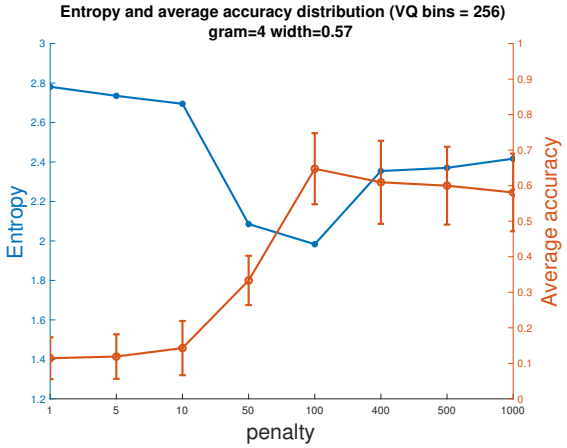
(a) Accuracy and entropy distribution for gram = 1



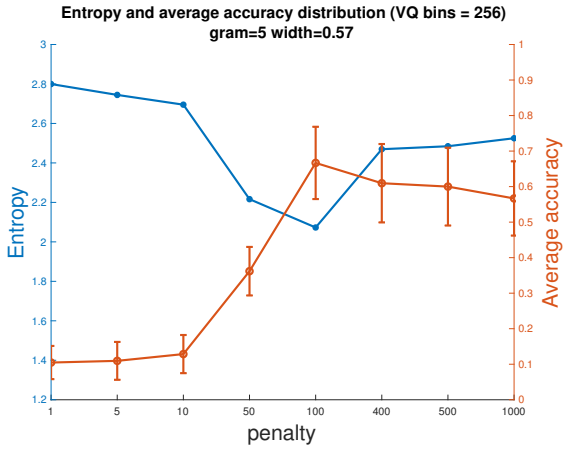
(b) Accuracy and entropy distribution for gram = 2



(c) Accuracy and entropy distribution for gram = 3

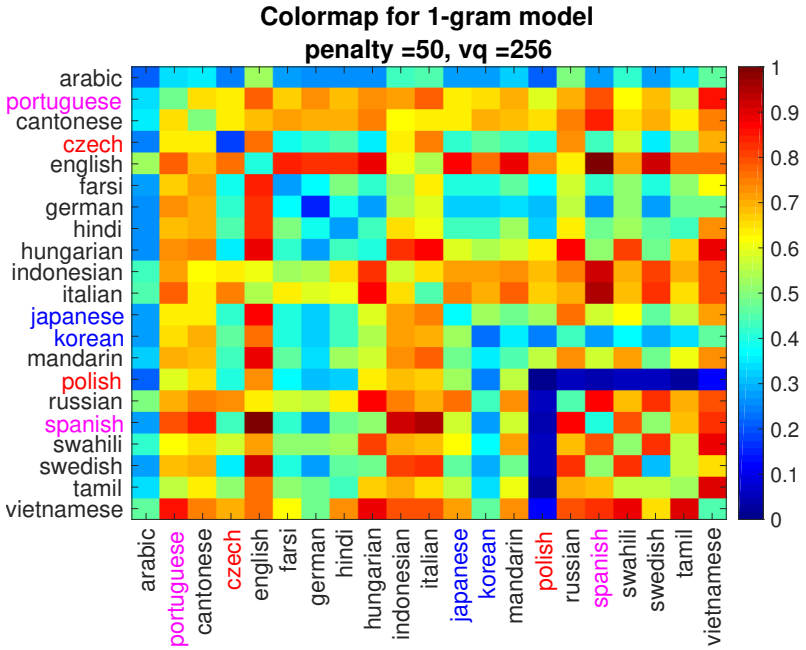


(d) Accuracy and entropy distribution for gram = 4

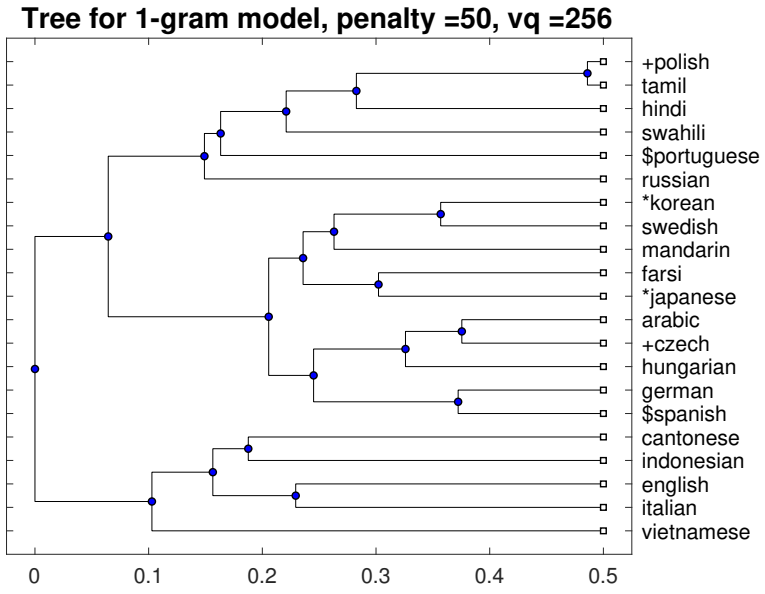


(e) Accuracy and entropy distribution for gram = 5

Figure 4.12: Accuracy and entropy distribution for n-grams. VQ bin size is 256 The x-axis is the penalty value. The left y-axis is the entropy value and the right y-axis is the accuracy value. The error bar on the average accuracy is the mean ± 2 standard error which obtains about 95% confidence interval of the estimate of the mean.



(a) Colour map of uni-gram



(b) Dendrogram of uni-gram

Figure 4.13: The 21 UNDHR audio language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 50 and the VQ bins is 256. Figure 4.13(a) shows the colour map of the language distance variations and Figure 4.13(b) shows the language tree which is built by the distances. The colour variation in Figure 4.13(b) shows the pairwise distances between languages.

4.3.8 Conclusion

As expected, the ALID results are worse than the TLID results. That said, the n-gram results are competitive with conventional ALID results, which is gratifying since the method we are using here is comparatively simple compared to other techniques. Figures 4.4 to 4.12 shows the distributions of entropy and accuracy with n-grams and penalties. The error bars are small which implies that the system is genuinely learning language distances rather than some proxy such as gender. As we explained in Section 4.3.6, the rules which are used to build the linguistic tree does not cover all of the language phoneme features. However, the n-gram model tries to compare all of the similarity and differences between the languages. It makes the language trees which are generated by the n-gram model looks random if we compare it to the linguistic language tree in Figure 2.7. Comparing with the TLID which uses the Unicode to represent the large character set, the IPA (Table 2.7) shows that ALID does not have thousands of phonemes variations like TLID characters. So, the features of phonemes are less distinctive than characters and it makes the distance entropy of ALID languages less than the TLID.

Looking into the entropies and colour maps, we find the large penalty improves the accuracy but the distances between languages are far from each other. A high value of distances matrix is not the best choice since we want the distances to show the relationships between languages which means, high distance variations. And also, a higher entropy may cause the low-order n-grams higher influence on distances than high-order n-grams. As vector quantisation is a kind of lossy compressor, the higher VQ bins, like 256, may not contain enough information for ALID even with high penalties. Thus it is obvious that an appropriate number of VQ bins improve the accuracy while making results worse if it is too small or too large.

4.4 Language distances calculated by compressor

In this section, we compute the audio language distances by using compressors. We applied the three compressors discussed in previous chapters: zip in 3.3.1.1, bzip in 3.3.2.1 and ppm in 3.3.3.1. We use the same feature extraction process as in 4.3 which vector quantizes the extracted MFCCs into Unicode characters. In zipping, we also wonder whether VQ binsize impacts on the results. In this case, we examine the compression results on 16, 32, 64, 128 and 256 bins, which is the same as n-gram method.

To evaluate and describe the audio results, and also to allow easy comparison with Cavnar and Trenkle [1994]'s ALID results, we use colour maps to show the pairwise distances between audio languages. The distance relationships are displayed by phylogenetic tree distances (explained in 3.2.2). Like audio Cavnar and Trenkle [1994]'s method, we use entropy to describe the distance distributions. We compare the recognition accuracy and entropy in each VQ bin case and compare the accuracy and entropy by the same compressor but with different VQ bins in Section 4.4.7.

4.4.1 Methods

The zipping methods we used in ALId is the same as in TLID (See section 3.3). However, there are slight differences in the data files, in other words, identical features to TLID. As we want to use the same method as TLID, we need to transform the waveform into strings. Thus, it is necessary to extract the waveform features by MFCCs and also convert them into Unicode characters by vector quantisation. The MFCCs features for zipping are generated by the same process as n-gram, which the feature extraction is explained in Section 4.2.1 and the vector quantisation is explained in Section 4.3.1.1. By applying Benedetto et al. [2002]'s zipping model, we calculate the language distances that are the relative entropies between languages.

For these results, a 0 following the name of the compressor denotes the interleaving status. For example, zip0, means non-interleaved string with zip compressor

and ppm1 means an interleaved string with a ppm compressor.

4.4.2 Language distance results with 16 bins

This section describes the language distance distribution by using colour map, phylogenetic tree and histogram distribution. The number of VQ bins is 16. The description of phylogenetic tree is in Section 3.2.2 and the description of histogram distribution is in Section 3.2.1.2. Figure 4.14 to 4.16 show the colour map of the languages distances. Figure 4.17 to 4.19 show the dendrogram of language distances.

Table 4.7: Entropy(top) and accuracy(bottom) values with histogram binwidth = 0.57, vq binsize = 16.

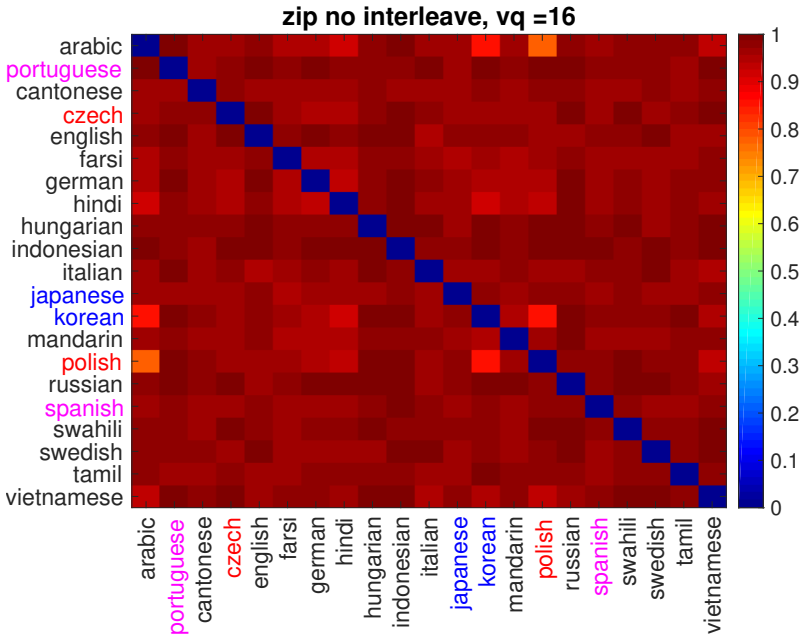
	Zppm0	Zppm1	Zzip0	Zzip1	Zbzip0	Zbzip1
Entropy	1.54	1.35	0.74	0.89	0.98	1.57
Accuracy	1	1	1	1	1	1

Table 4.7 concludes the entropy values of the histogram distribution for ppm, zip and bzip with interleaved and non-interleaved data. The results show the recognition accuracies of all compressions are 100% and the highest entropy is 1.54. According to Equation 3.10, the distances of the languages are calculated by analysing the compressed length of the strings. For measuring the distance of language itself, the zipping method compresses one string with itself. So the ppm, bzip and zip do not need to predict the characters which have never been seen before. Thus, the compression entropy of language itself is always the smallest and the recognition accuracy is always 100%. For reference, a histogram with two equiprobable bins would have an entropy of 1 bit whereas a 16-bin histogram with equiprobable bins would have an entropy of 4 bits. Thus 1.54 bits indicates a very non-smooth histogram (an all-or-nothing distance).

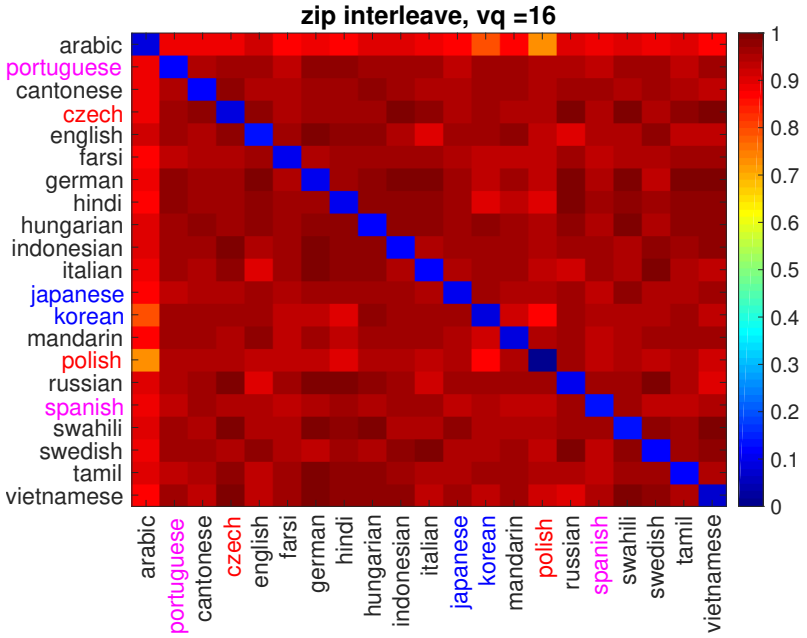
Figure 4.14 to Figure 4.19 show the colour map and dendrogram of the pairwise language distances for each compression with interleaved and non-interleaved data. As we previously mentioned in Section 4.3.3, the 16 VQ bins characters exist nearly all languages, which means the strings contain many continuing and repeated

characters. So that the interleaving does not significantly impact on the entropy for zip and ppm since for a fixed length of buffer string, zip (LZ77) sequentially calculates the maximum repetitions and ppm uses the Markov chain for calculating probabilities by predicting next character. Interleaving impacts on bzip is because the Burrow-Wheeler transform sorts the characters by frequencies and the run-length encoding shortens the length of the encoding. For example, supposing the bzip blocksize is 6, there is a buffer string of $\boldsymbol{a}$ = "bnnaaa" and a buffer string of $\boldsymbol{b}$ = "aaabbb", the interleaved string of $\boldsymbol{a}$ and $\boldsymbol{b}$ is $\boldsymbol{s}$ = "banana|", the "|" stands for the end of buffer. For ppm and zip, they compress string by the order of characters. However, bzip (See Section 3.3.2.1) firstly process "bananaa|" into "annb|aa" that tends to put the same characters together. So for a fixed length of buffer string, bzip gets better compressibility for languages shared longer repeated characters(like "aa" and "nn" in the string s), which makes the interleaved results show a higher entropy than the non-interleaved.

The dendrograms show a poor language grouping: the Indo-Hittite language family is randomly located over the trees. Although Figure 4.17(b) shows Spanish and Portuguese are close, it is linguistically impossible that they are the same origin as Japanese. According to the colour map, it is also obvious that under the 16 VQ bins, although the languages are close to themselves, the distance variations are not easy to observe and the entropy is much lower than Cavnar and Trenkle [1994]'s n-gram model.

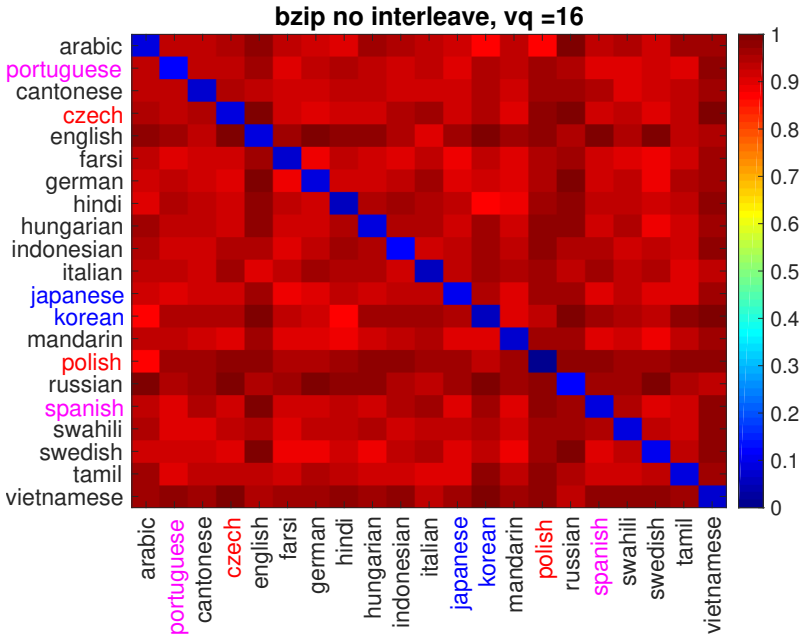


(a) without interleave

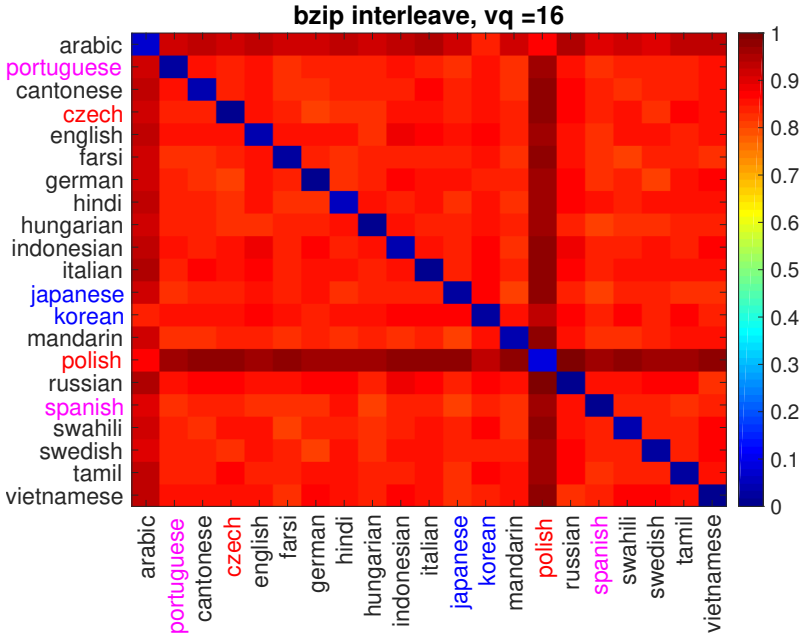


(b) with interleave

Figure 4.14: The 21 UNDHR audio languages distances are computed by zip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 4.14(a) shows the non-interleaved result and Figure 4.14(b) shows the interleaved result.

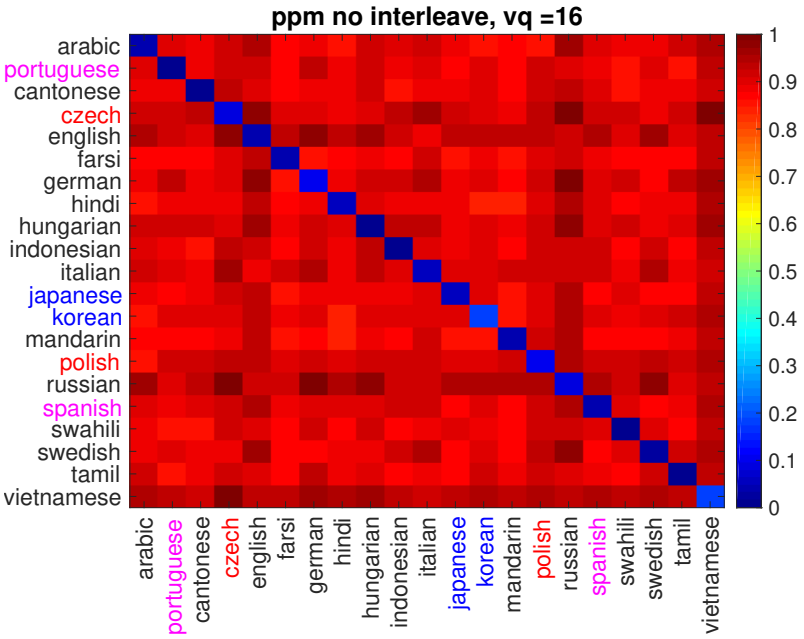


(a) without interleave

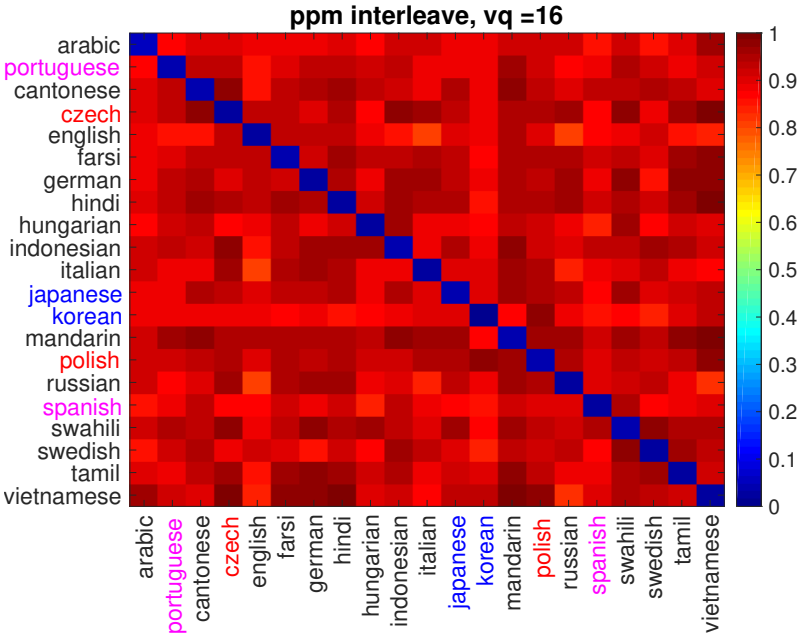


(b) with interleave

Figure 4.15: The 21 UNDHR audio languages distances are computed by bzip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 4.14(a) shows the non-interleaved result and Figure 4.15(b) shows the interleaved result.

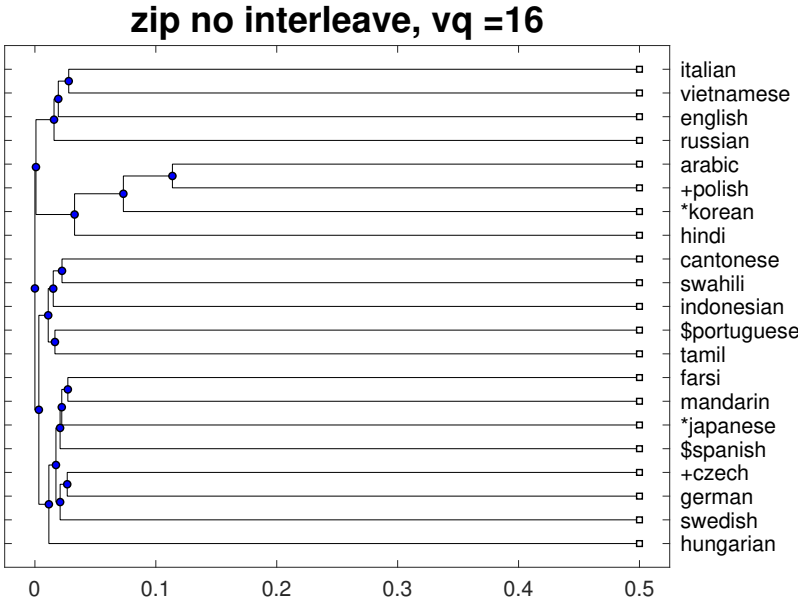


(a) without interleave

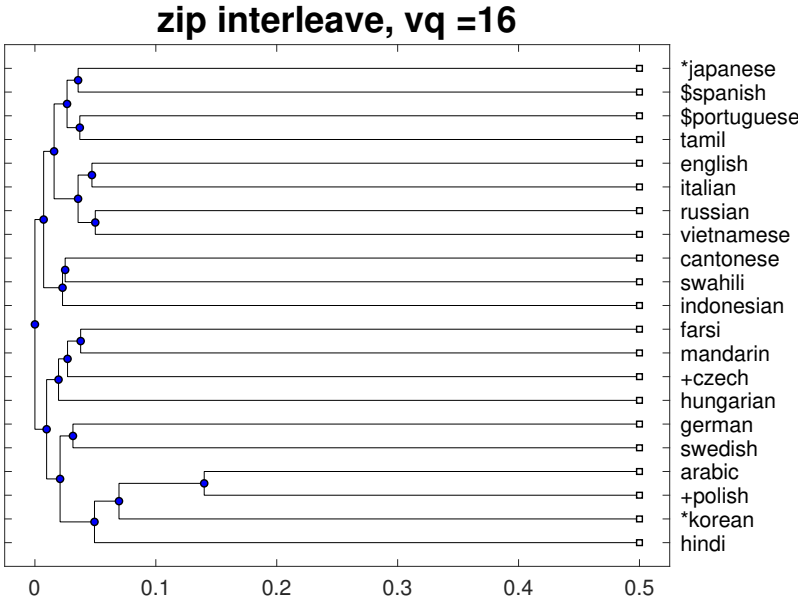


(b) with interleave

Figure 4.16: The 21 UNDHR audio languages distances are computed by ppm and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 4.16(a) shows the non-interleaved result and Figure 4.16(b) shows the interleaved result.

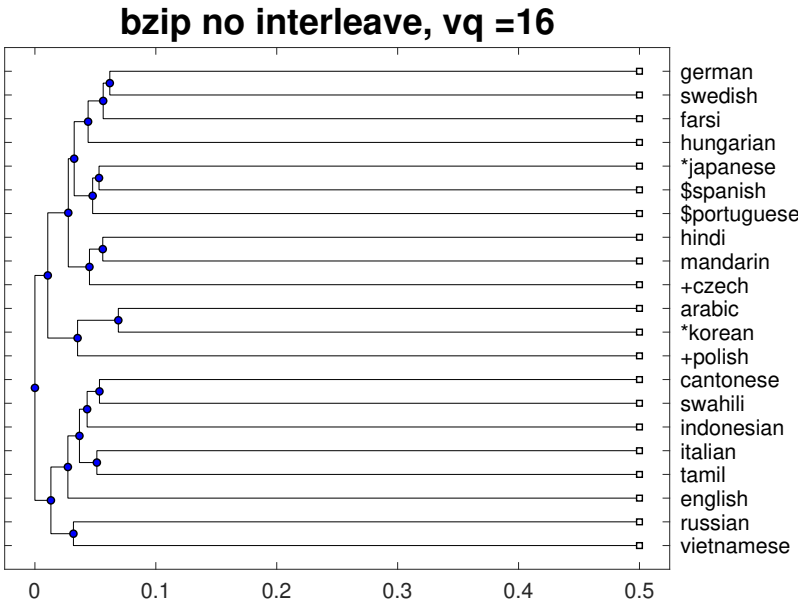


(a) without interleave

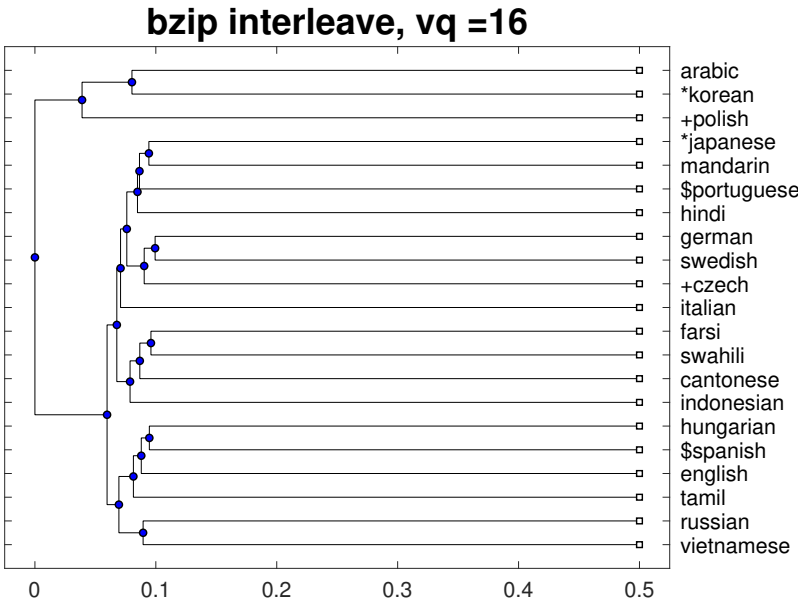


(b) with interleave

Figure 4.17: The 21 UNDHR audio languages distances are computed by zip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 4.17(a) shows the non-interleaved result and Figure 4.17(b) shows the interleaved result.

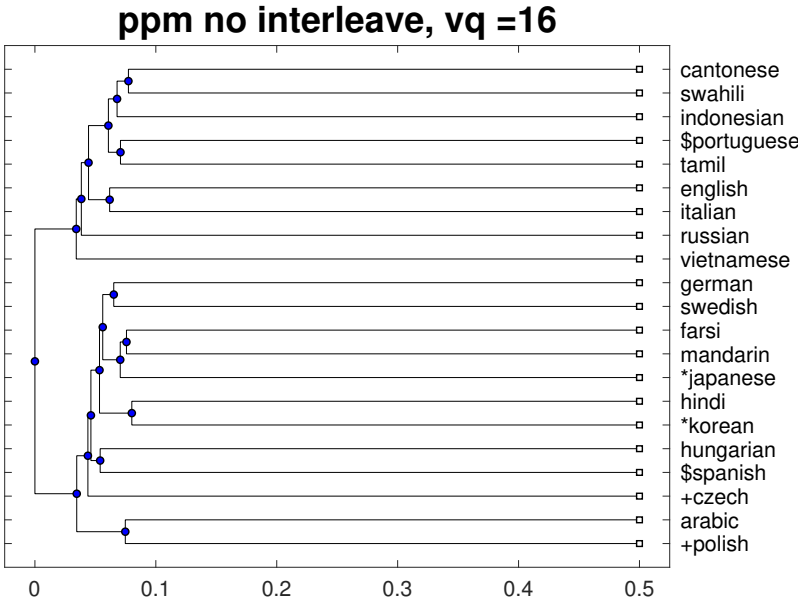


(a) without interleave

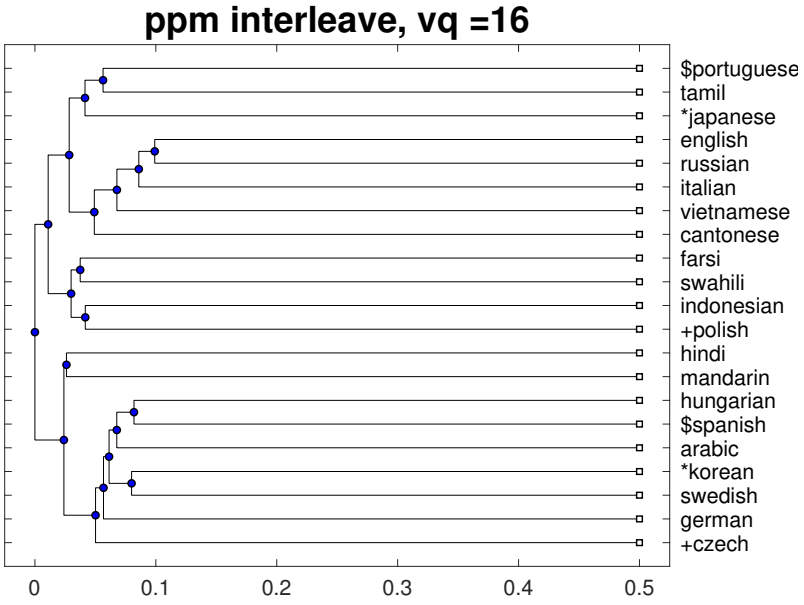


(b) with interleave

Figure 4.18: The 21 UNDHR audio languages distances are computed by bzip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 4.18(a) shows the non-interleaved result and Figure 4.18(b) shows the interleaved result.



(a) without interleave



(b) with interleave

Figure 4.19: The 21 UNDHR audio languages distances are computed by ppm and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 4.19(a) shows the non-interleaved result and Figure 4.19(b) shows the interleaved result.

4.4.3 Language distance results with 32 bins

This section displays the results with 32 VQ bins. Figure 4.20 to 4.22 show the colour map of the languages distances and Figure 4.23 to 4.25 show the dendrogram of language distances. The diagrams are produced the same as 16 VQ bins in Section 4.4.2 but with 32 VQ bins.

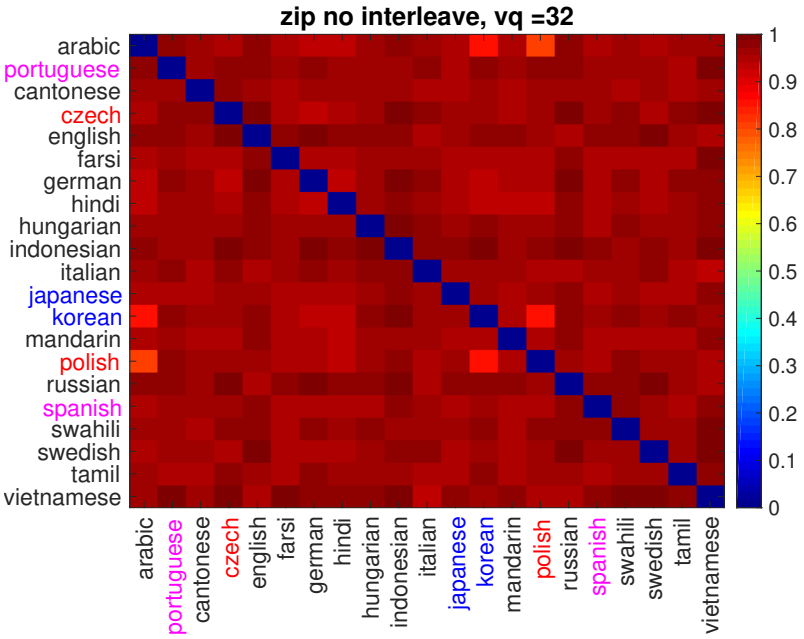
Table 4.8: Entropy values which histogram binwidth = 0.57 and the VQ binsize = 32 .

	Zppm0	Zppm1	Zzip0	Zzip1	Zbzip0	Zbzip1
Entropy	0.99	1.48	0.72	1.05	1.06	0.79
Accuracy	1	1	1	1	1	1

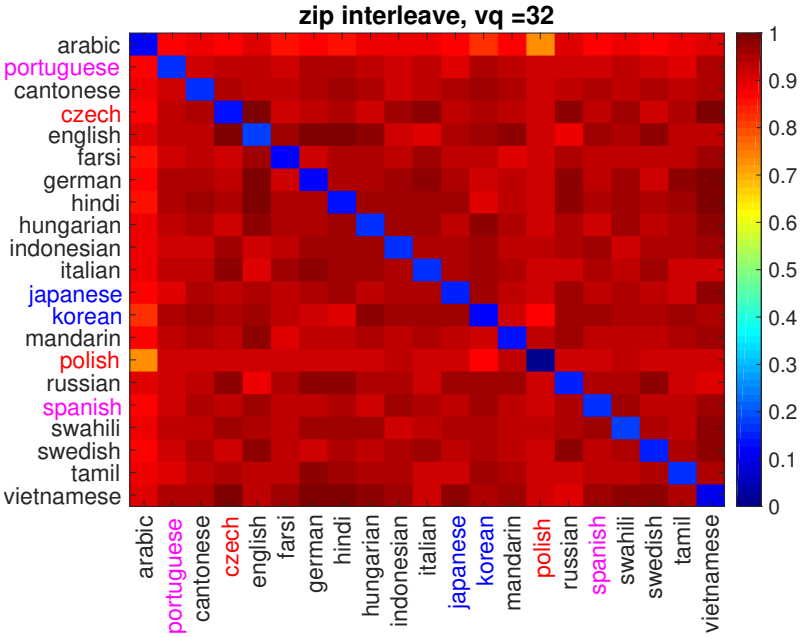
Table 4.8 concludes the entropy values of the histogram distribution between the pairwise distances of languages for ppm, zip and bzip with the interleaved and the non-interleaved data. The results show the recognition accuracies of all compressions are 100% and the highest entropy is ppm with interleaved, which is 1.48. Figure 4.20 to Figure 4.25 show the colour map and dendrogram of the pair-wise language distances for each compression (interleaved and non-interleaved). As in the 32 VQ bins, languages share a part of the character set, the ppm and zip interleaved entropy results are higher than the non-interleaved because of the unseen characters. For bzip, in the fixed length of buffer string, the diversity of characters in one language increased but the interleaved string might have longer repeated strings than non-interleaved. For example, if there are two strings $\boldsymbol{a}$ =“aabbccddc” and $\boldsymbol{b}$ = “abcccdeed”, the size of buffer string is 8. The interleaved string is $\boldsymbol{i}$ = “aaabbccc|cddedecd” and the non-interleaved string is $\boldsymbol{n}$= “aabbccddc|abcccdeed”. Since Burrow-Wheeler transform groups repeated characters, the interleaved string gets better zipping performance. Thus, the entropy of non-interleaved results is higher than the interleaved.

The colour maps perform more pairwise distance variations (the distance entropy) than the 16 VQ bins, especially for Figure 4.21(b), the bzip interleaved case. However, the dendrogram still describes it is a poor grouping. Figure 4.20(a) shows

that for the interleaved and the non-interleaved data with zip, the Indo-Hittites are randomly located in all subtrees. Also, there is no evidence that Japanese is close to Korean. Figure 4.22(a) and 4.22(a) show the same problem. In this case, we can conclude that zipping methods with 32 bins cannot show the relationships between the languages.

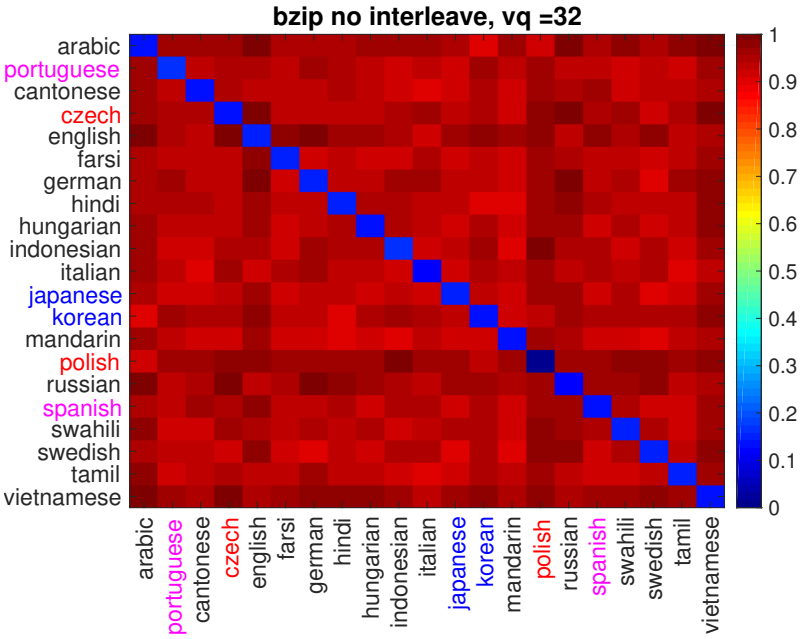


(a) without interleave

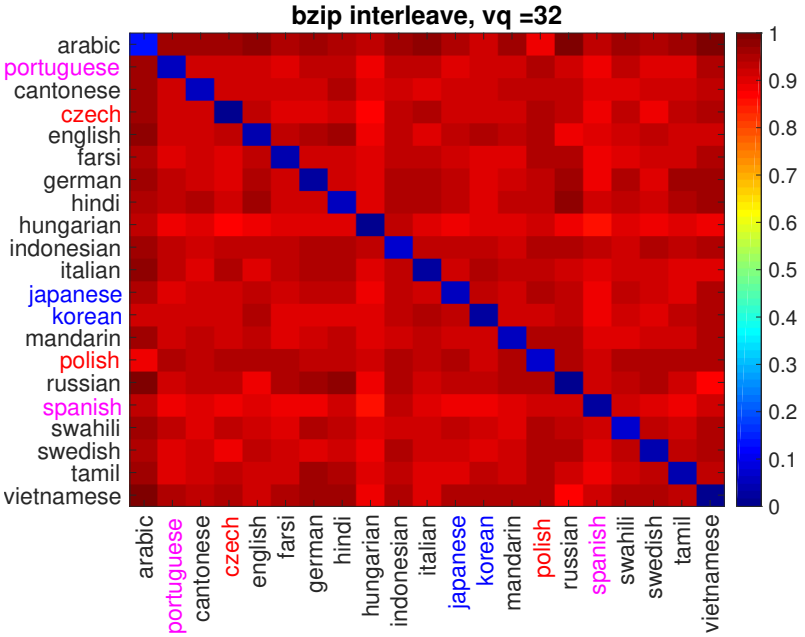


(b) with interleave

Figure 4.20: The 21 UNDHR audio languages distances are computed by zip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 4.20(a) shows the non-interleaved result and Figure 4.20(b) shows the interleaved result.

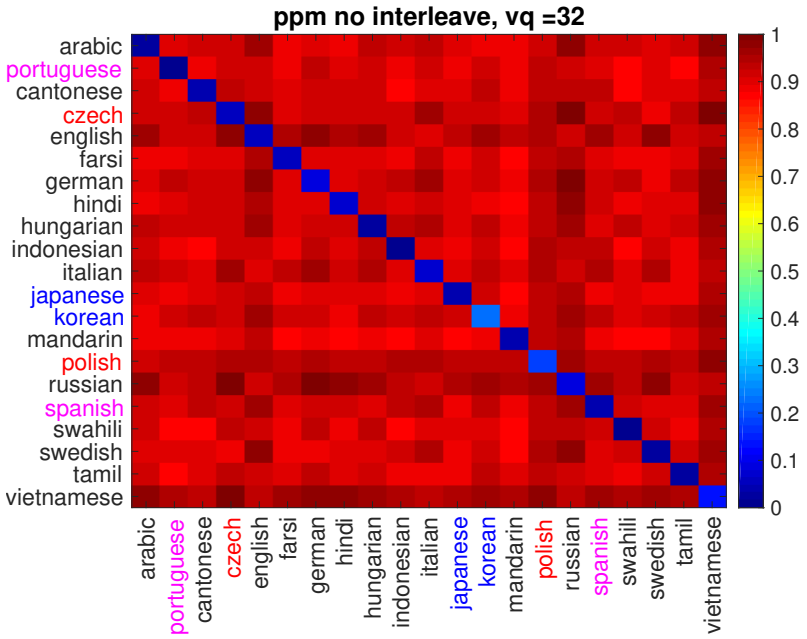


(a) without interleave

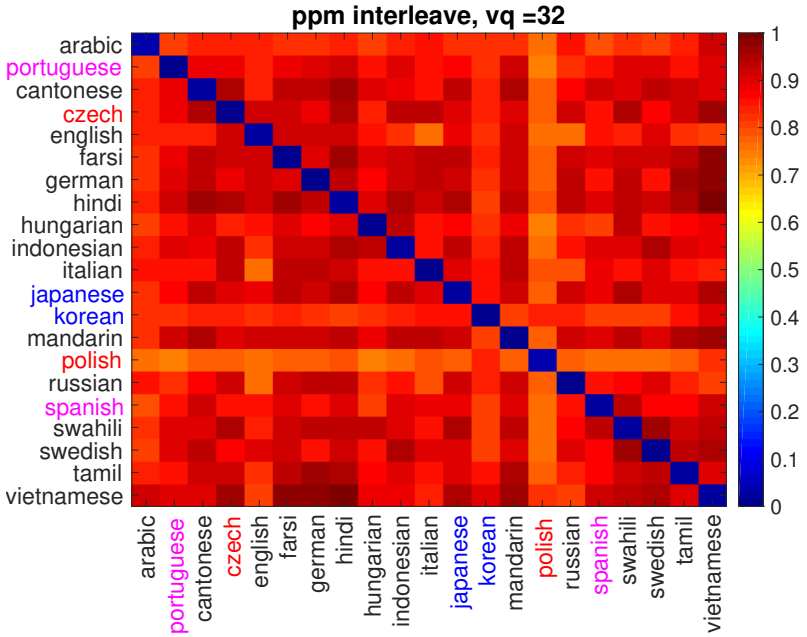


(b) with interleave

Figure 4.21: The 21 UNDHR audio languages distances are computed by bzip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 4.20(a) shows the non-interleaved result and Figure 4.21(b) shows the interleaved result.

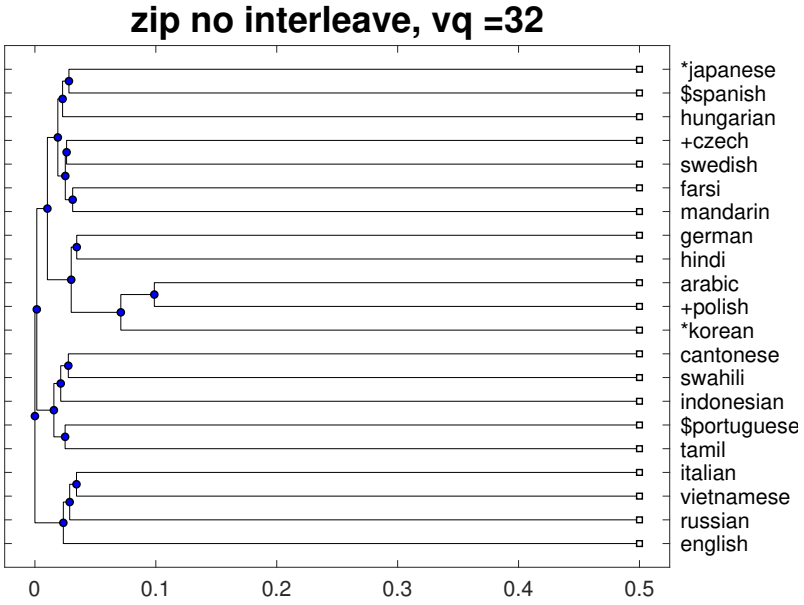


(a) without interleave

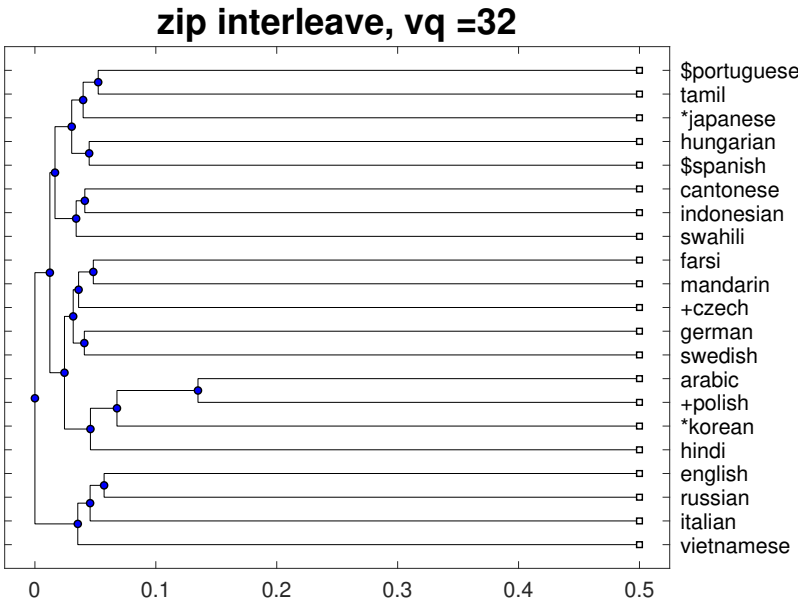


(b) with interleave

Figure 4.22: The 21 UNDHR audio languages distances are computed by ppm and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 4.22(a) shows the non-interleaved result and Figure 4.22(b) shows the interleaved result.

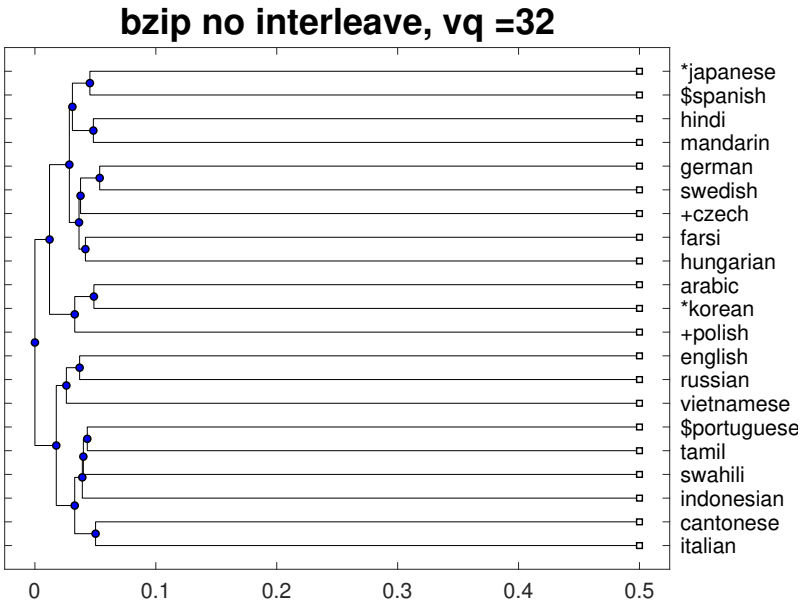


(a) without interleave

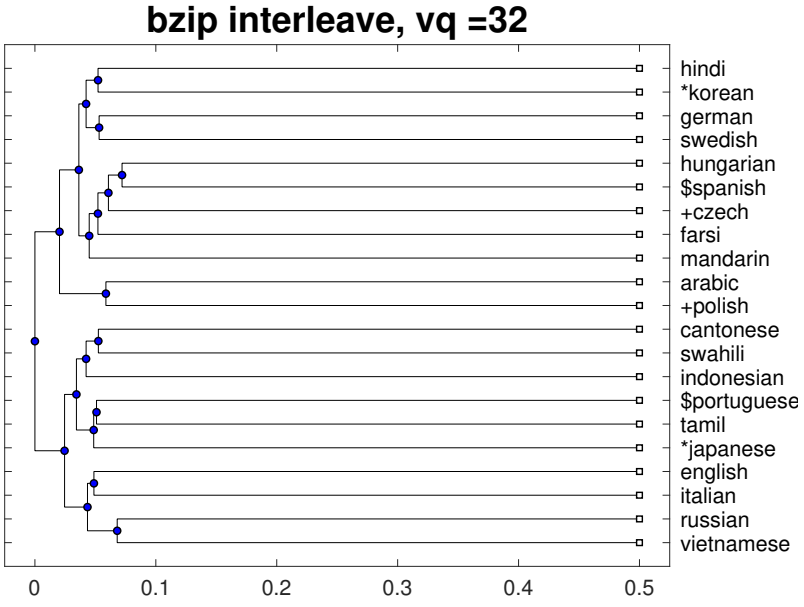


(b) with interleave

Figure 4.23: The 21 UNDHR audio languages distances are computed by zip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 4.23(a) shows the non-interleaved result and Figure 4.23(b) shows the interleaved result.

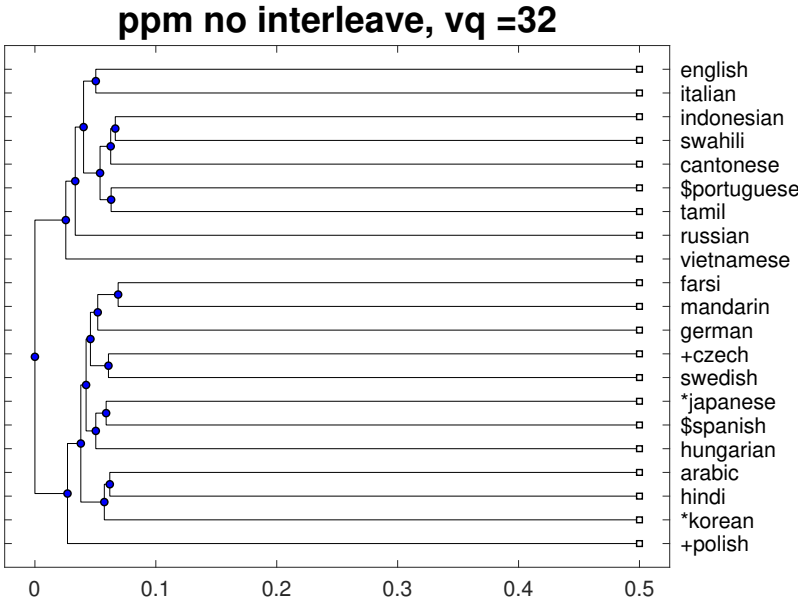


(a) without interleave

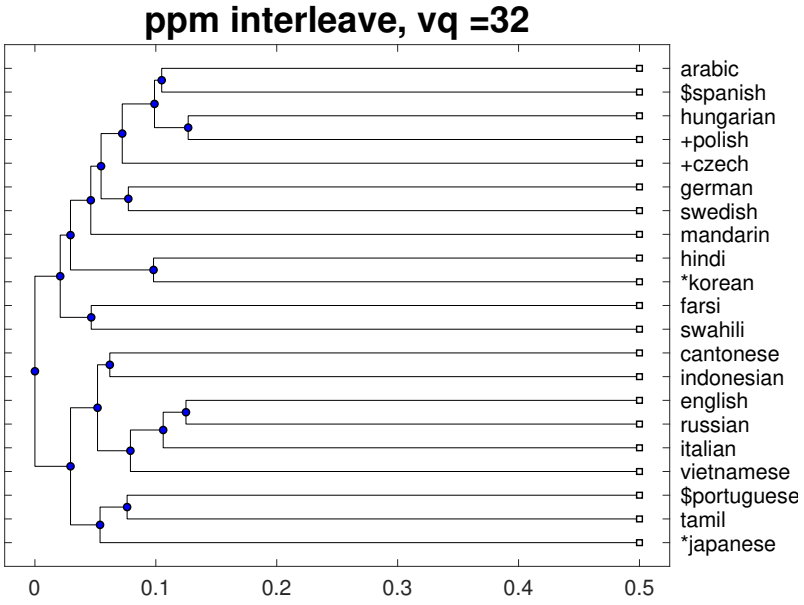


(b) with interleave

Figure 4.24: The 21 UNDHR audio languages distances are computed by bzip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 4.24(a) shows the non-interleaved result and Figure 4.24(b) shows the interleaved result.



(a) without interleave



(b) with interleave

Figure 4.25: The 21 UNDHR audio languages distances are computed by ppm and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 4.25(a) shows the non-interleaved result and Figure 4.25(b) shows the interleaved result.

4.4.4 Language distance results with 64 bins

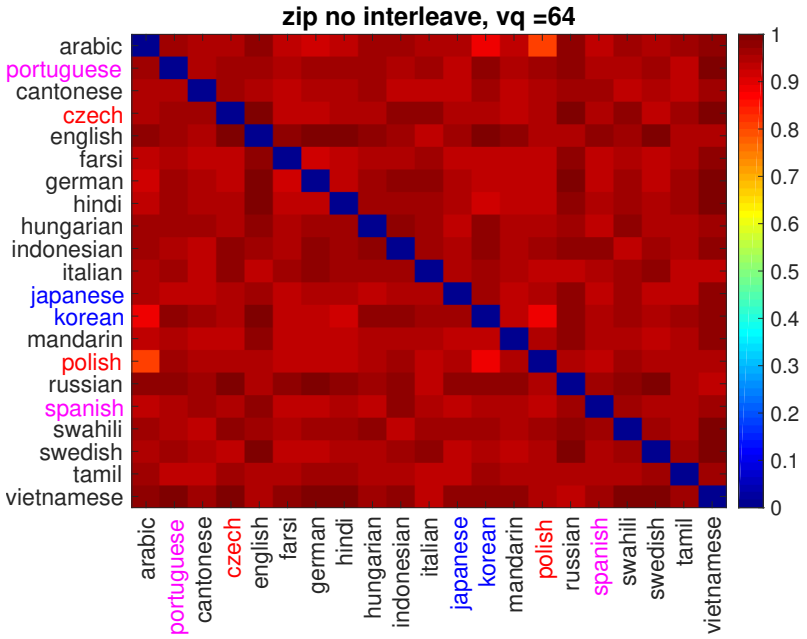
This section displays the results with 64 VQ bins. Figure 4.26 to 4.28 show the colour map of the languages distances and Figure 4.29 to 4.31 show the dendrogram of language distances. The diagrams are produced the same as 16 VQ bins in Section 4.4.2 but with 64 VQ bins.

Table 4.9: Entropy values which histogram binwidth = 0.57 and the VQ binsize = 64 .

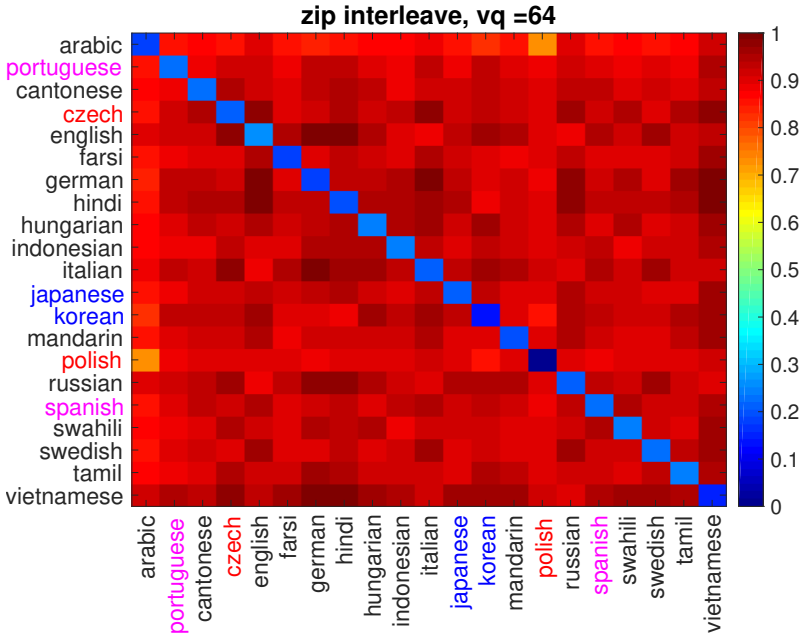
	Zppm0	Zppm1	Zzip0	Zzip1	Zbzip0	Zbzip1
Entropy	1.20	1.40	0.72	1.16	0.69	1.48
Accuracy	1	1	1	1	1	1

Table 4.9 concludes the entropy values of the histogram distribution between the pair-wise distances of languages for ppm, zip and bzip with interleaved and non-interleaved data. The results show the recognition accuracies of all compressions are 100% and the highest entropy is bzip with interleaved, which is 1.48. Figure 4.26 to Figure 4.31 show the colour map and dendrogram of the pair-wise language distances for each compression (interleaved and non-interleaved). As the 64 VQ bins case has a larger character set, the ppm and zip methods get a higher entropy for interleaved than non-interleaved. And for 64 bins, bzip also perform a higher entropy for interleaved data than non-interleaved. This tells us the repeated characters do not have a high occurrence so bzip has similar compressibility to ppm and zip.

The colour maps of Figure 4.26 to 4.28 perform some distances variation like 32 VQ bin case. We still can find languages are close to themselves but the relationships with other languages are also not so clear. The structure helps us to find the relationships but the Indo-Hittite languages are still randomly in different subtrees. Since we can not conclude the language relationships based on the 64 VQ bins, we then focus on 128 VQ bins.

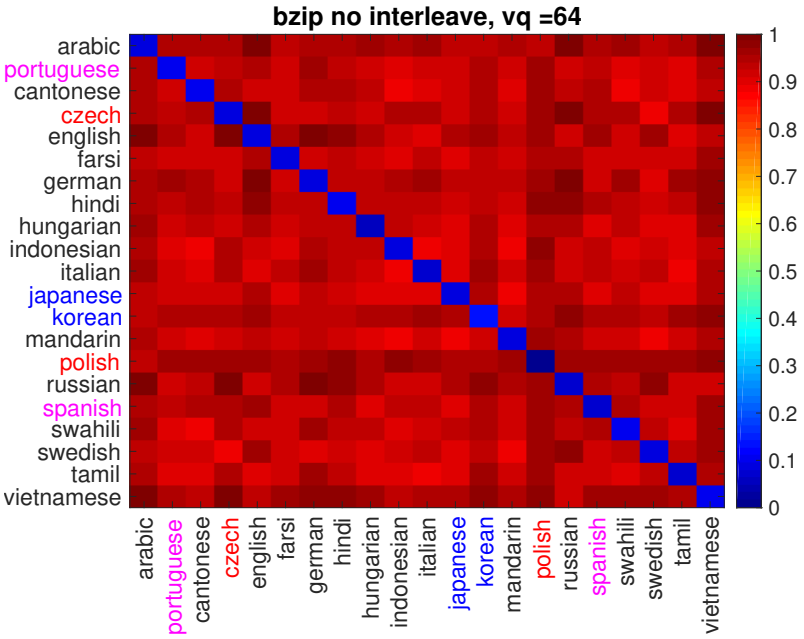


(a) without interleave

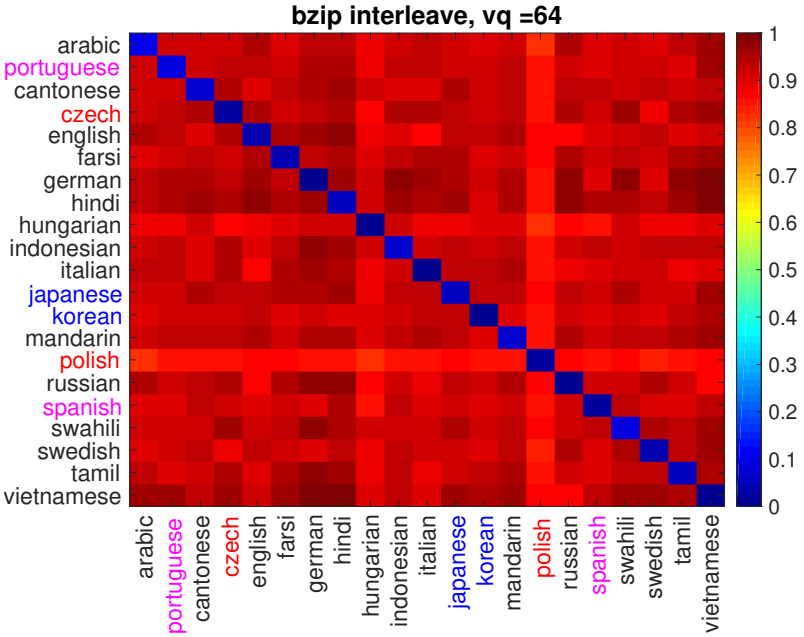


(b) with interleave

Figure 4.26: The 21 UNDHR audio languages distances are computed by zip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 4.26(a) shows the non-interleaved result and Figure 4.26(b) shows the interleaved result.

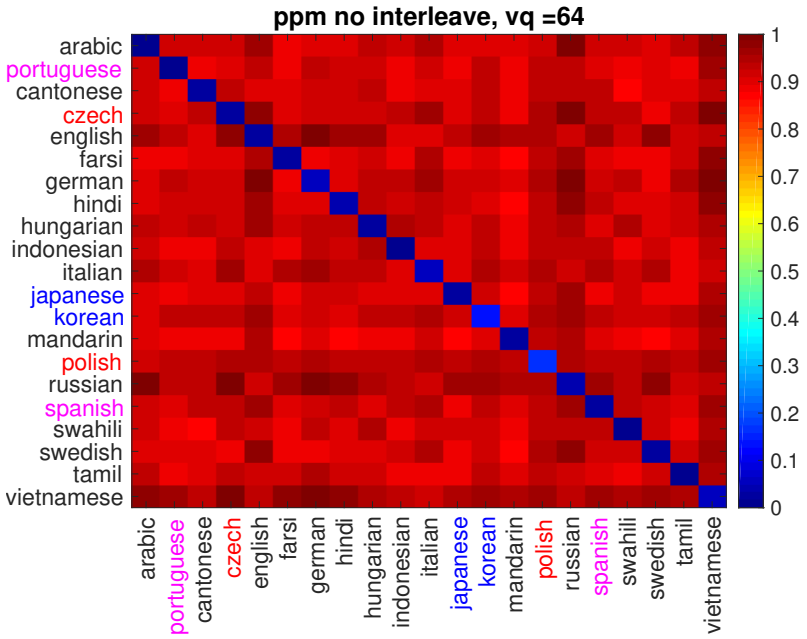


(a) without interleave

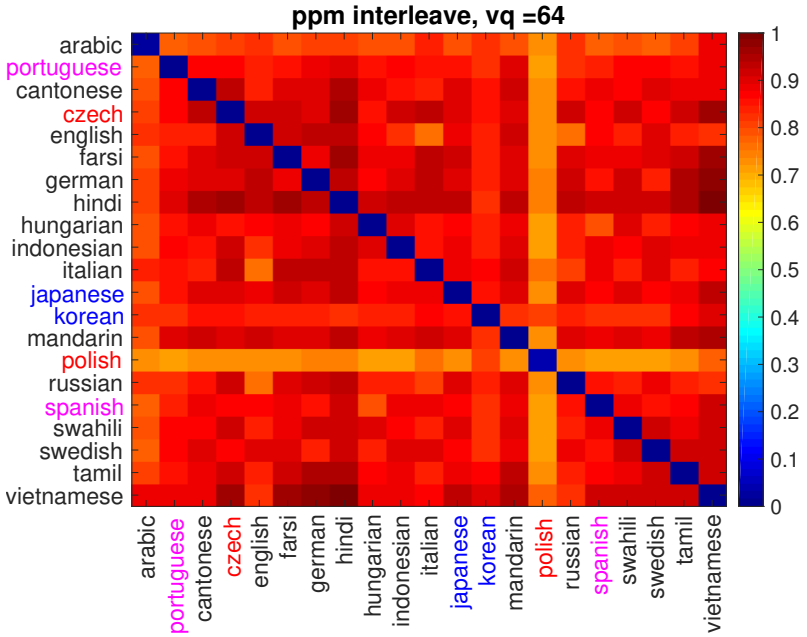


(b) with interleave

Figure 4.27: The 21 UNDHR audio languages distances are computed by bzip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 4.26(a) shows the non-interleaved result and Figure 4.27(b) shows the interleaved result.

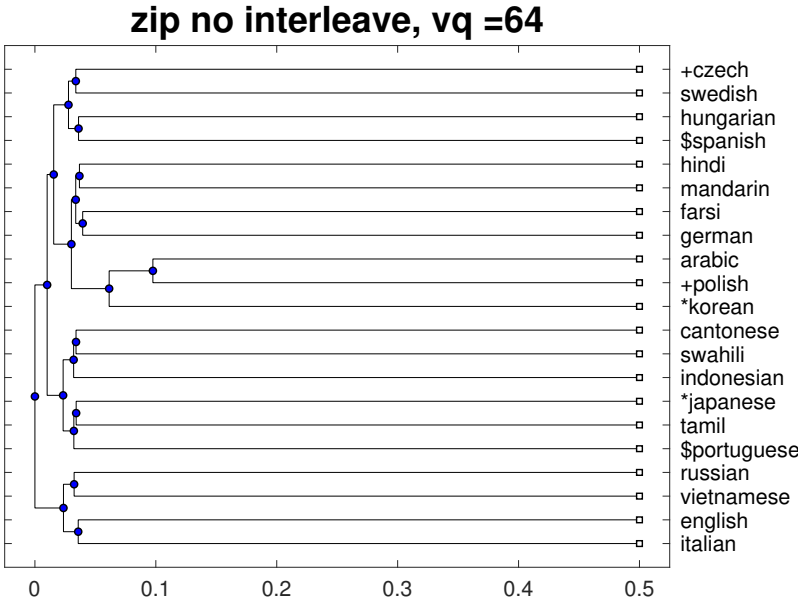


(a) without interleave

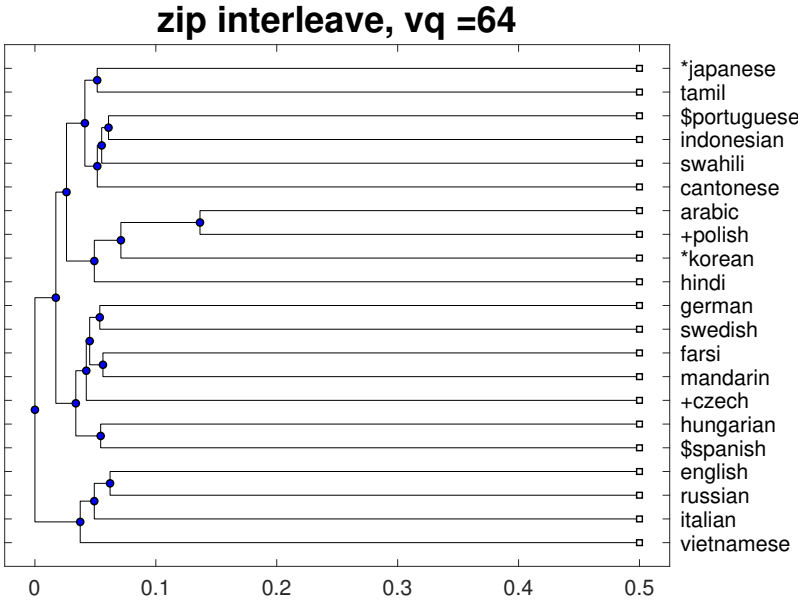


(b) with interleave

Figure 4.28: The 21 UNDHR audio languages distances are computed by ppm and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 4.28(a) shows the non-interleaved result and Figure 4.28(b) shows the interleaved result.

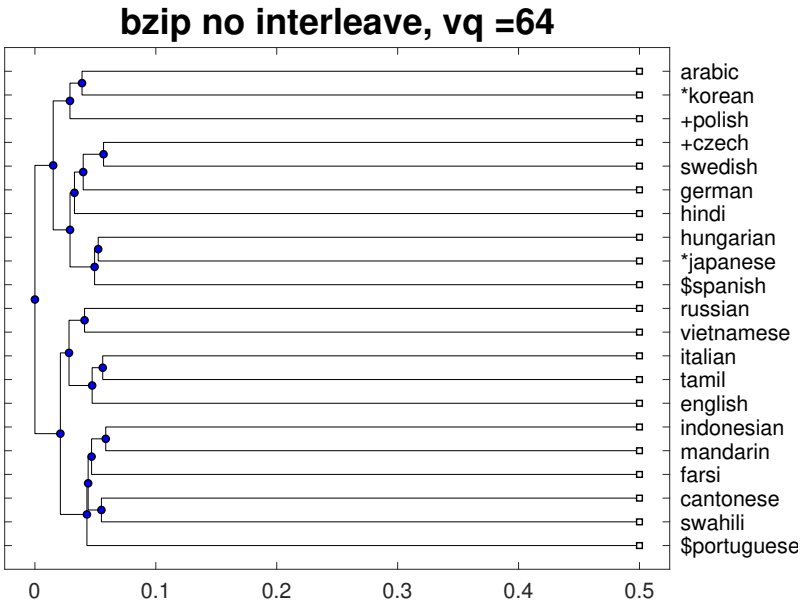


(a) without interleave

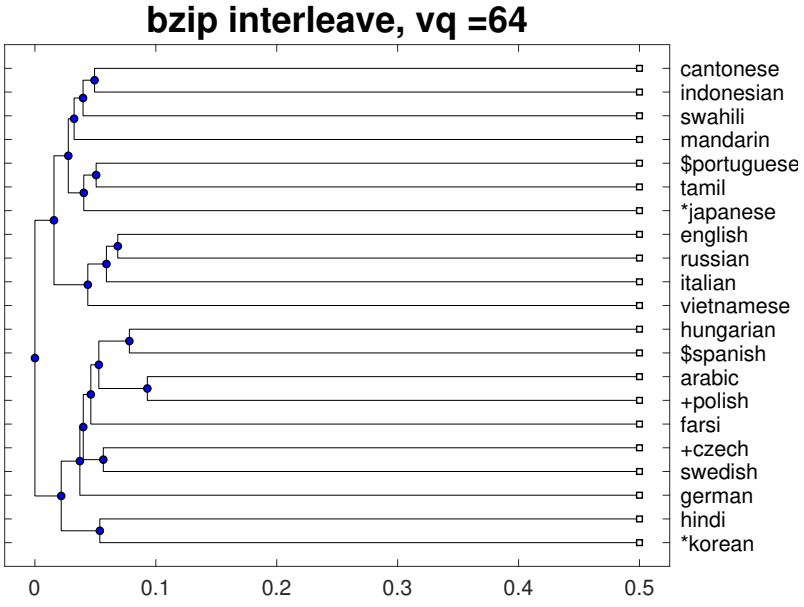


(b) with interleave

Figure 4.29: The 21 UNDHR audio languages distances are computed by bzip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 4.30(a) shows the non-interleaved result and Figure 4.30(b) shows the interleaved result.

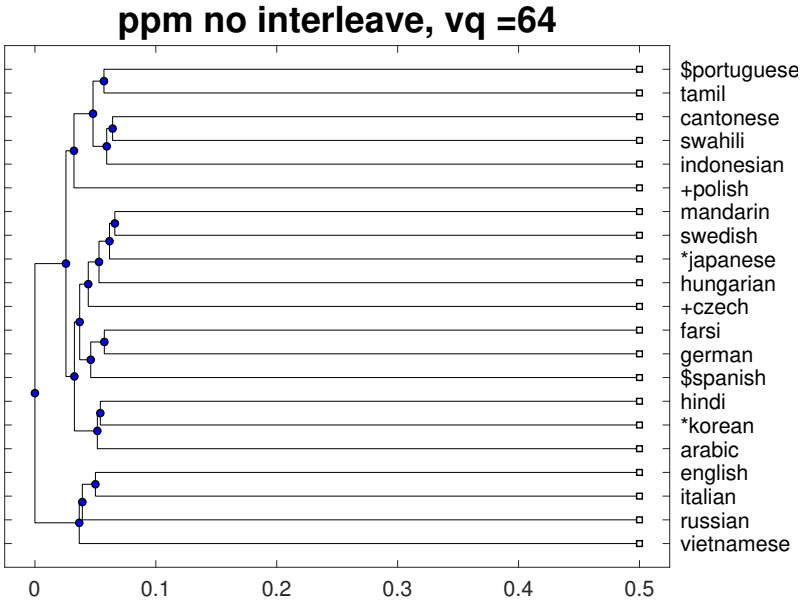


(a) without interleave

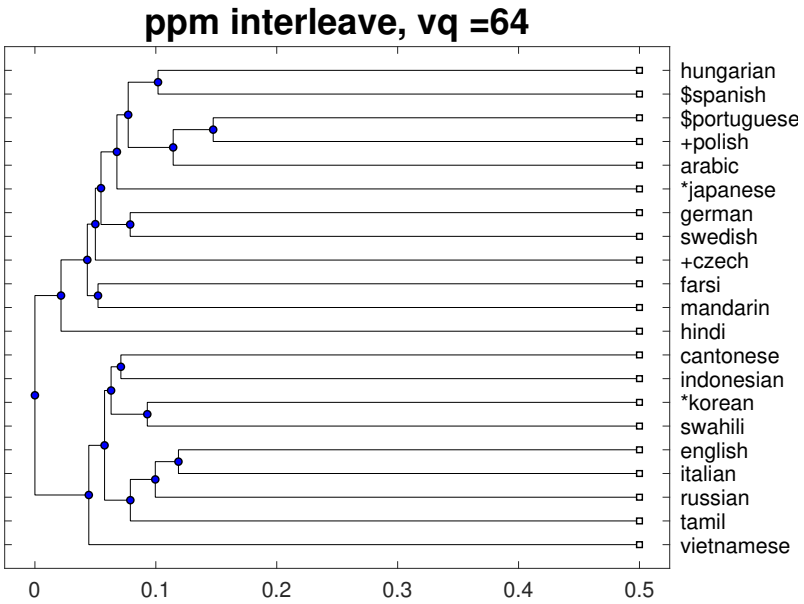


(b) with interleave

Figure 4.30: The 21 UNDHR audio languages distances are computed by bzip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 4.30(a) shows the non-interleaved result and Figure 4.30(b) shows the interleaved result.



(a) without interleave



(b) with interleave

Figure 4.31: The 21 UNDHR audio languages distances are computed by ppm and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 4.31(a) shows the non-interleaved result and Figure 4.31(b) shows the interleaved result.

4.4.5 Language distance results with 128 bins

This section displays the results with 128 vector quantisation bins. Figure 4.32 to 4.34 show the colour map of the languages distances. Figure 4.35 to 4.37 show the dendrogram of language distances. The diagrams are produced the same as 16 VQ bins in Section 4.4.2 but with 128 VQ bins.

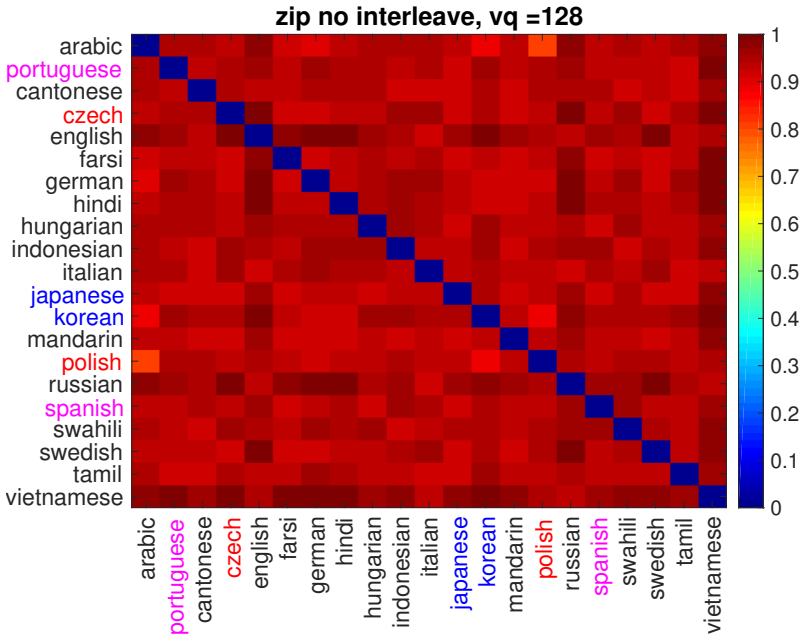
Table 4.10 concludes the entropy values of histogram distribution between pair-wise distances of languages for ppm, zip and bzip with interleaved and non-interleaved data. The results show the recognition accuracies of all compressions are 100% and the highest entropy is zip with interleaved, which is 1.49. Like section 4.4.4, the entropy results of interleaved are all higher than non-interleaved because of the big size of the character set.

Table 4.10: Entropy values which histogram binwidth = 0.57 and the VQ binsize = 128 .

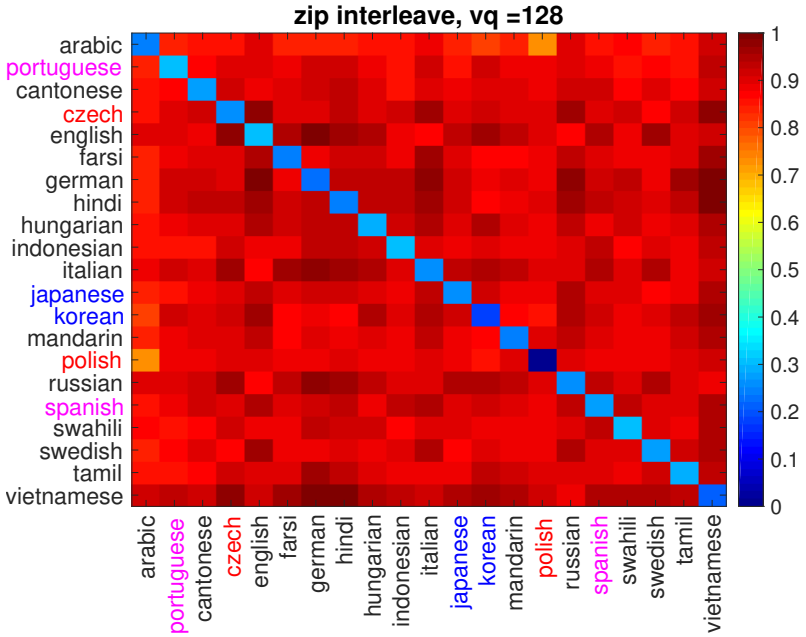
	Zppm0	Zppm1	Zzip0	Zzip1	Zbzip0	Zbzip1
Entropy	0.65	1.41	0.76	1.49	1.02	1.20
Accuracy	1	1	1	1	1	1

Figure 4.32 to Figure 4.37 show the colour map and the dendrogram of the pair-wise language distances for each compression (interleaved and non-interleaved). For colour maps, Figure 4.32 to Figure 4.34 shows all zip methods get 100% accuracy that correspond to the Table 4.10. However, it is still hard to find the language distances relationships between languages. So we investigate the dendrogram in Figure 4.35, 4.36 and 4.37. Although the entropies of 128 VQ bins show that all zip methods compress languages based on the character variation instead of long, repeated characters, the language is still not linguistically well grouped as what we expect. In Figure 4.35(a), it is still hard to find the language classes. Japanese and Korean are related to different Indo-Hittite languages rather than each other. The Indo-Hittite languages are not close to each other and randomly located in the tree. Also, for Figure 4.35(b), there is no evidence in linguist and geography that Swedish is close to Japanese rather than other Indo-Hittite languages. Spanish

and Portuguese are in different language subtrees which means, their distances are far more than most of the distances between other languages. Also there is no clear background truth can claim that Polish and Korean have the same origin. Figure 4.36(a) shows the same problem that the distances between Spanish and Portuguese describe that bzip with interleaved languages in 128 VQ bins is a bad language classification in Indo-Hittite languages. For ppm, Figure 4.37(b) describes that Japanese is close to Portuguese rather than Korean. Also, there is no Indo-Hittite language is well grouped into one subtree except English, Italian and Russian. However, the language distance between Polish and Czech is far from each other than other languages which is a similar problem as Figure 4.35(b). Looking into the interleaved case for ppm (Figure 4.37(a)), although the Portuguese and Polish are close since they are all Indo-Hittite language, Spanish should not be close to Japanese and Hungarian rather than Portuguese.

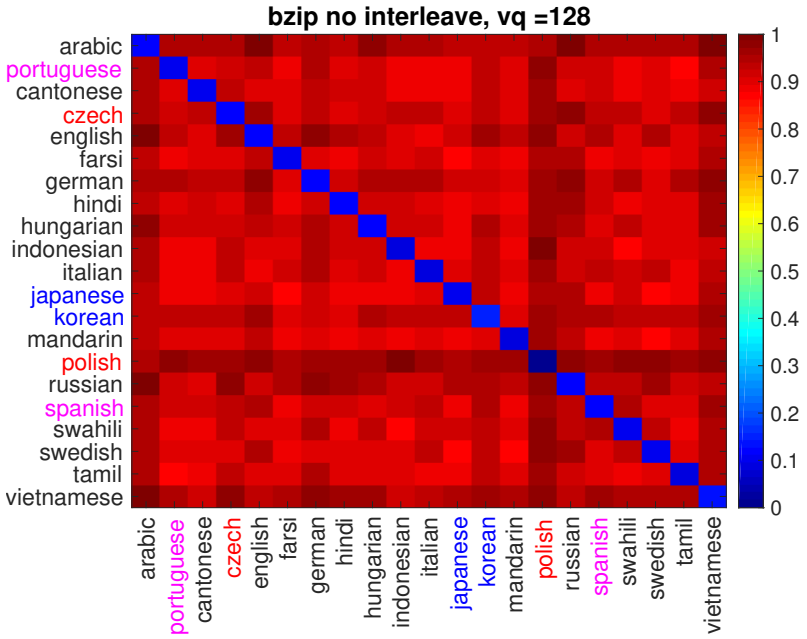


(a) without interleave

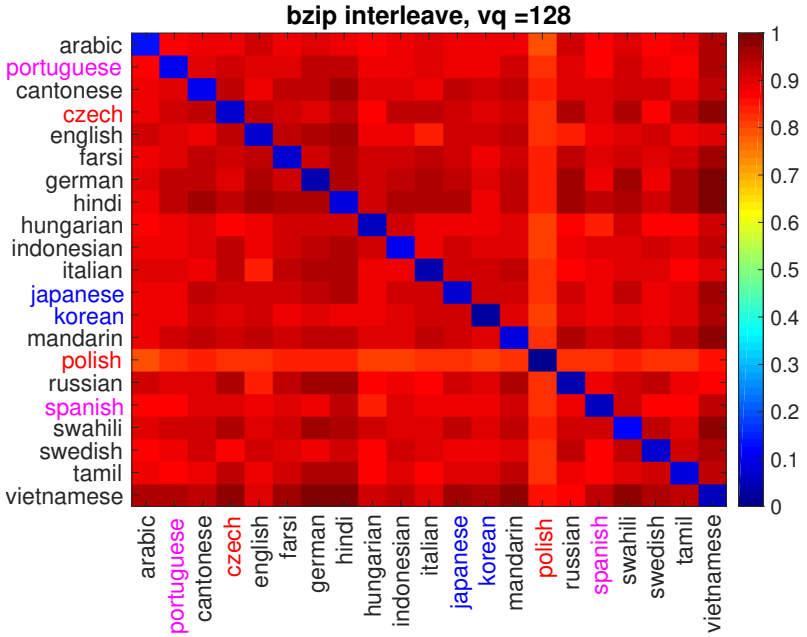


(b) with interleave

Figure 4.32: The 21 UNDHR audio languages distances are computed by zip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 4.32(a) shows the non-interleaved result and Figure 4.32(b) shows the interleaved result.

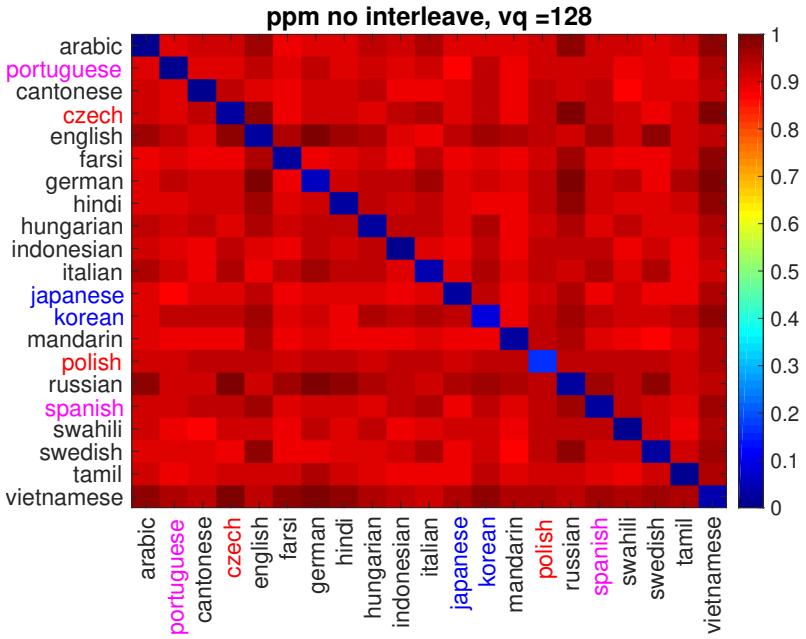


(a) without interleave

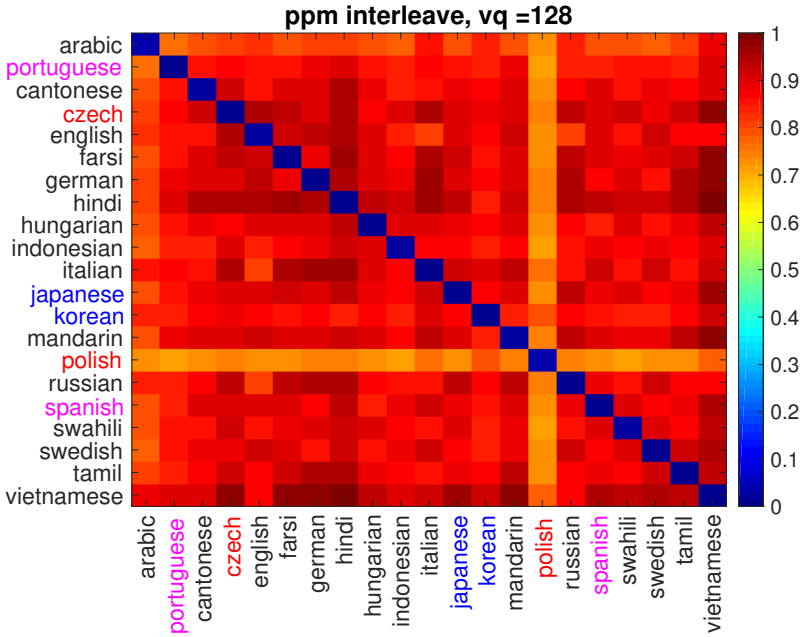


(b) with interleave

Figure 4.33: The 21 UNDHR audio languages distances are computed by bzip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 4.32(a) shows the non-interleaved result and Figure 4.33(b) shows the interleaved result.

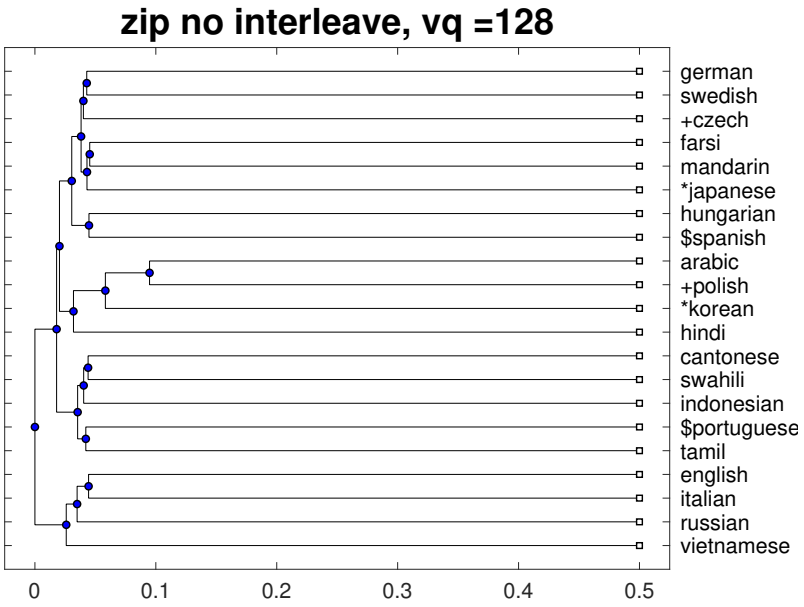


(a) without interleave

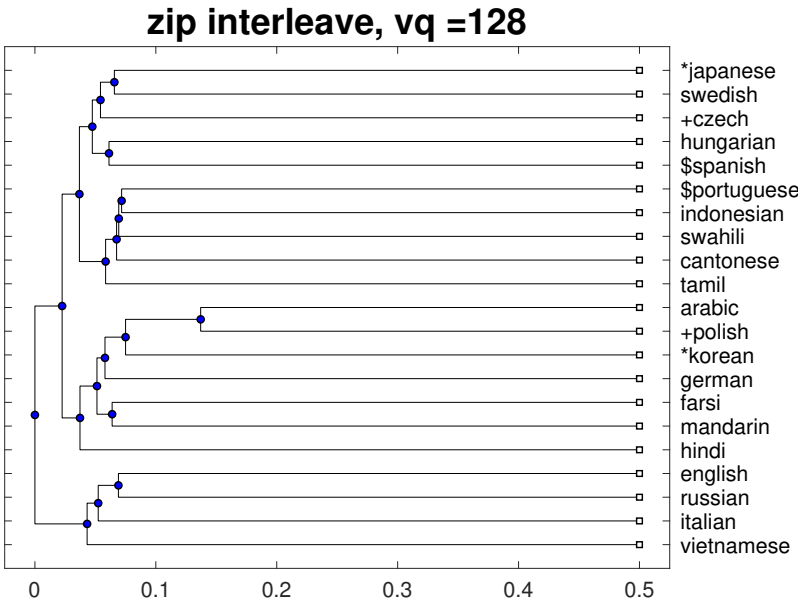


(b) with interleave

Figure 4.34: The 21 UNDHR audio languages distances are computed by ppm and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 4.34(a) shows the non-interleaved result and Figure 4.34(b) shows the interleaved result.

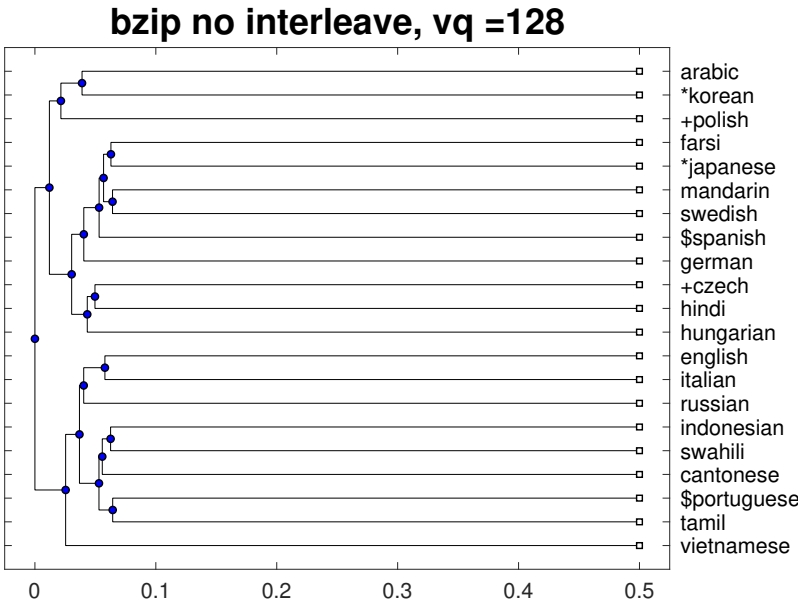


(a) without interleave

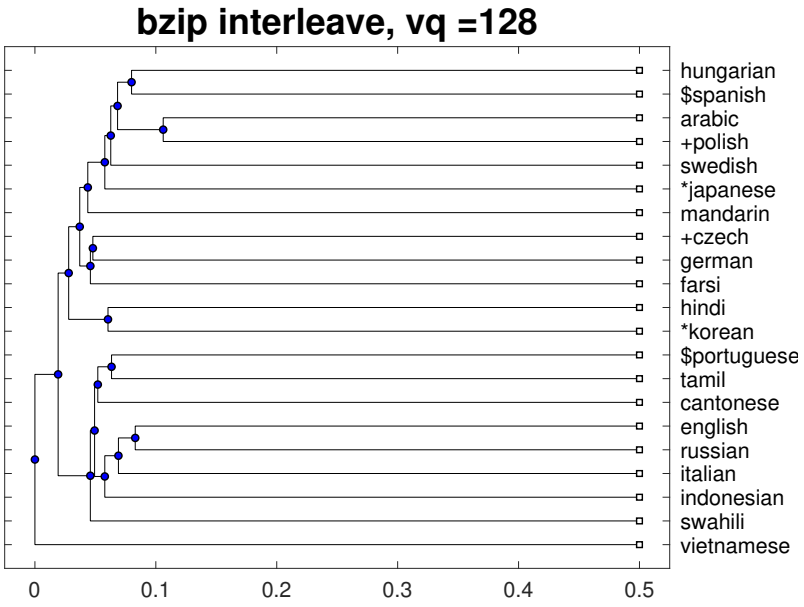


(b) with interleave

Figure 4.35: The 21 UNDHR audio languages distances are computed by zip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 4.35(a) shows the non-interleaved result and Figure 4.35(b) shows the interleaved result.

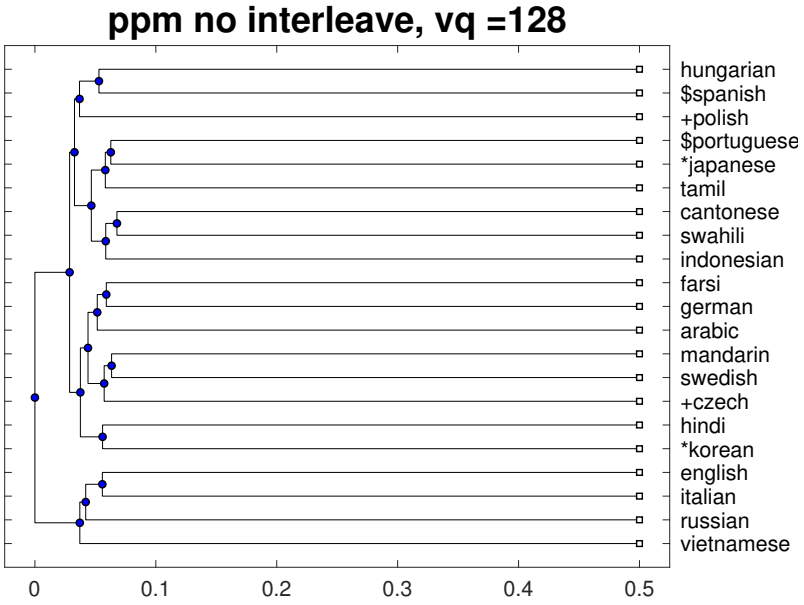


(a) without interleave

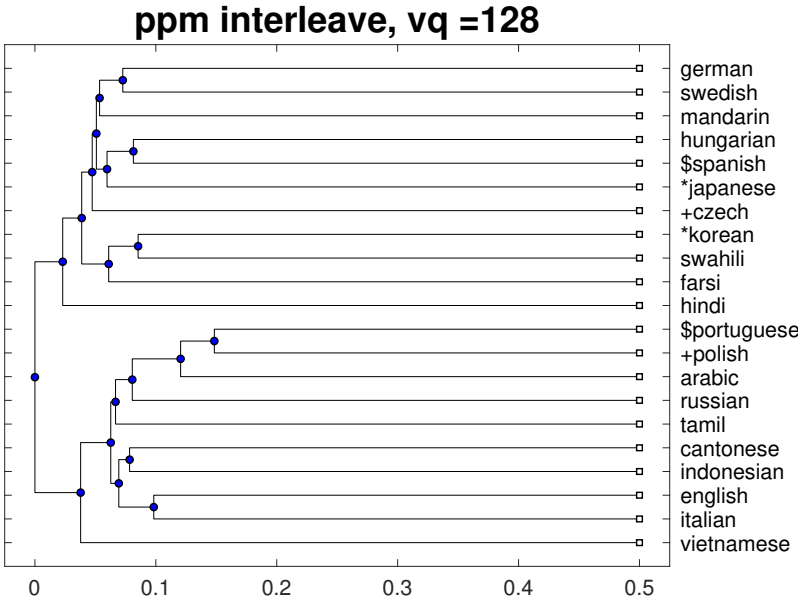


(b) with interleave

Figure 4.36: The 21 UNDHR audio languages distances are computed by bzip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 4.36(a) shows the non-interleaved result and Figure 4.36(b) shows the interleaved result.



(a) without interleave



(b) with interleave

Figure 4.37: The 21 UNDHR audio languages distances are computed by ppm and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 4.37(a) shows the non-interleaved result and Figure 4.37(b) shows the interleaved result.

4.4.6 Language distance results with 256 bins

This section displays the results with 256 vector quantisation bins. Figure 4.38 to 4.40 shows the colour map of the languages distances. Figure 4.41 to 4.43 shows the dendrogram of language distances. The diagrams are produced the same as 16 VQ bins in Section 4.4.2 but with 256 VQ bins.

Table 4.11: Entropy values which histogram binwidth = 0.57 and the VQ binsize = 256 .

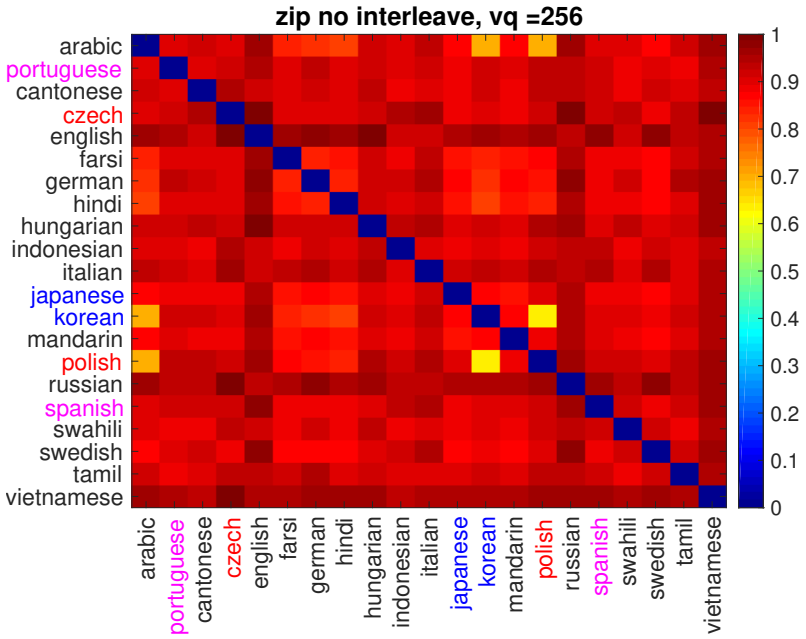
	Zppm0	Zppm1	Zzip0	Zzip1	Zbzip0	Zbzip1
Entropy	0.64	1.59	1.33	1.77	1.29	1.09
Accuracy	1	1	1	1	1	1

Table 4.11 concludes the entropy values of the histogram distribution between the pair-wise distances of languages for ppm, zip and bzip with interleaved and non-interleaved data. The results show the recognition accuracies of all compressions are 100% and the highest entropy is zip with interleaved, which is 1.77. The ppm and zip produce a higher entropy in interleaved case but bzip with non-interleaved result shows a higher entropy than interleaved. As we have mentioned in ALID n-gram results (Section 4.3.7), VQ is a lossy compressor which cause 256 VQ bins loses more information. In zipping, it may cause the languages to share more Unicode characters with other languages. As we previously mentioned in 32 VQ bins (Section 4.4.3), the bzip uses BWT and RLE to compress the duplicated characters which decrease the entropy value of the interleaved result.

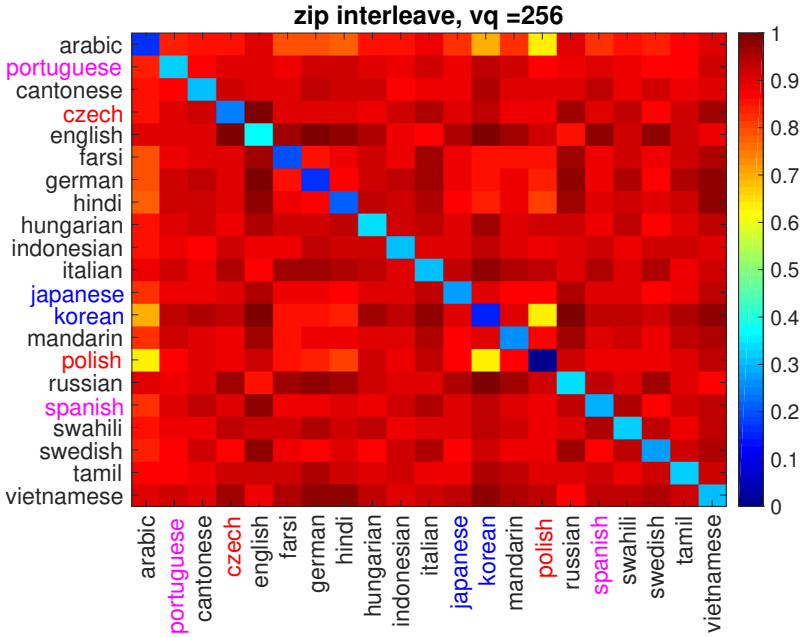
Figure 4.38 to Figure 4.43 show the colour map and dendrogram of the pair-wise language distances for each compression (interleaved and non-interleaved). For colour maps, Figure 4.38 to Figure 4.40 shows all zip methods get 100% accuracy that correspond to the Table 4.11. For zip diagrams (Figure 4.38), both interleaved and non-interleaved results tell that Polish and Korean are close to Arabic and also Polish and Korean are close to each other. As we know in the language tree, the Indo-Hittite languages should have closer distances, thus there is no linguistic reason that Polish is far from other Indo-Hittite languages but is close to Korean. It

proves that the 256 VQ bins lose information and confuses the language classification results. Since the other language distances are not easy to observe, we investigate the dendrogram in Figure 4.41, 4.42 and 4.43.

In Figure 4.41(a), it is still hard to find the language classes. The distance relationship between Polish, Korean and Arabic is also mentioned and prove that zip does not group language properly. Also, this problem shows a bad language group and make results unreliable in Figure 4.39(a), 4.39(b), 4.40(a) and 4.40(a) since there is no evidence in linguistics and geography that Swedish is close to Japanese rather than other Indo-Hittite languages. For this reason, we can conclude that 256 is not appropriate to find the language relationships.

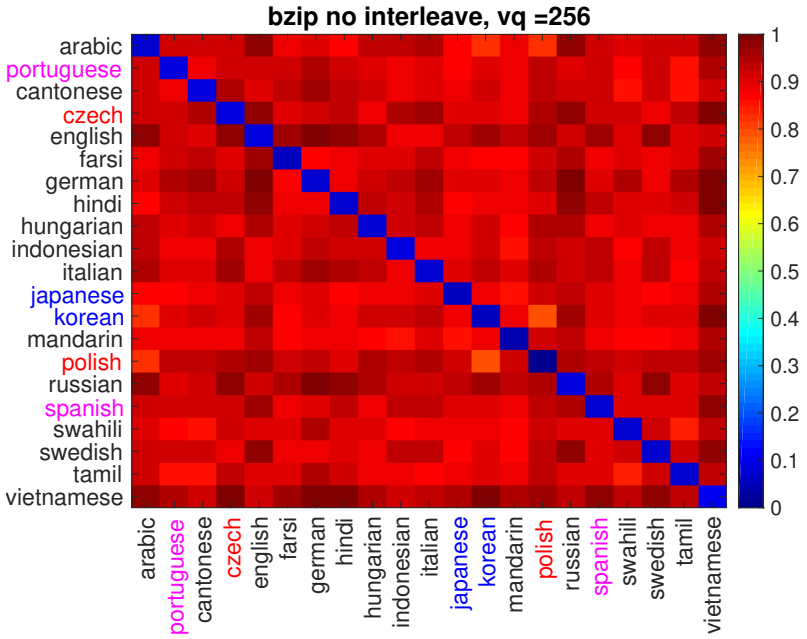


(a) without interleave

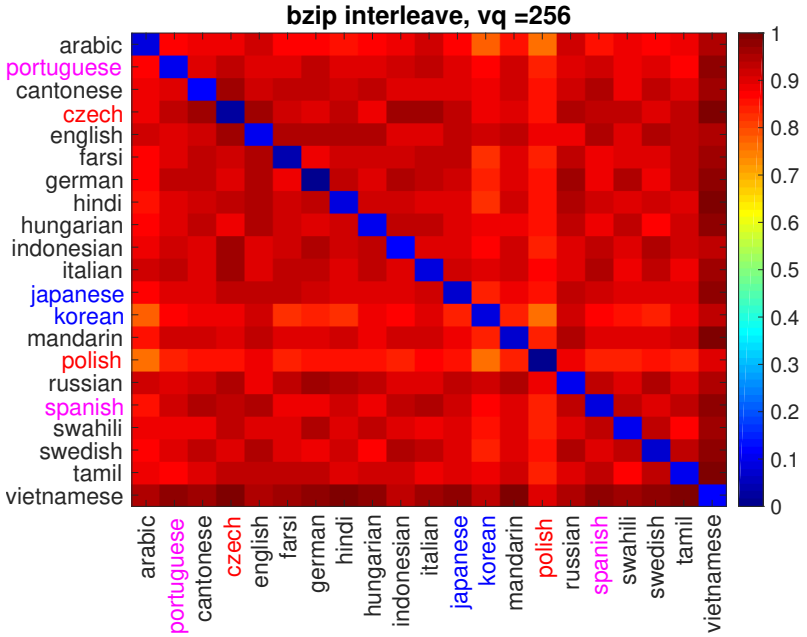


(b) with interleave

Figure 4.38: The 21 UNDHR audio languages distances are computed by zip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 256. Figure 4.38(a) shows the non-interleaved result and Figure 4.38(b) shows the interleaved result.

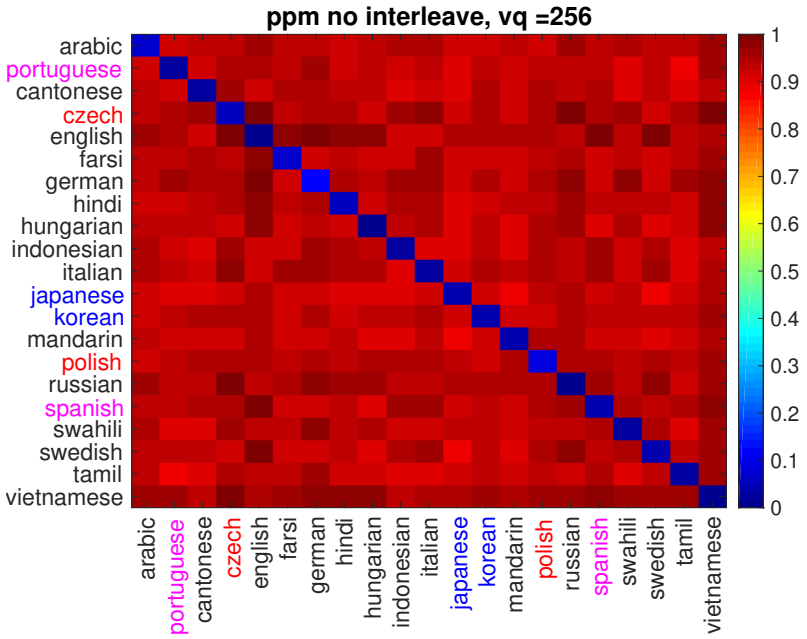


(a) without interleave

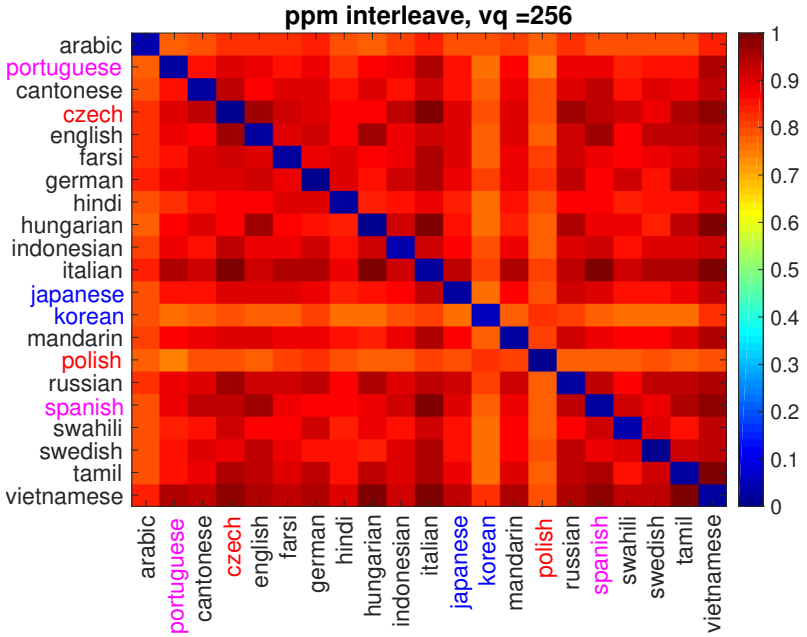


(b) with interleave

Figure 4.39: The 21 UNDHR audio languages distances are computed by bzip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 256. Figure 4.38(a) shows the non-interleaved result and Figure 4.39(b) shows the interleaved result.

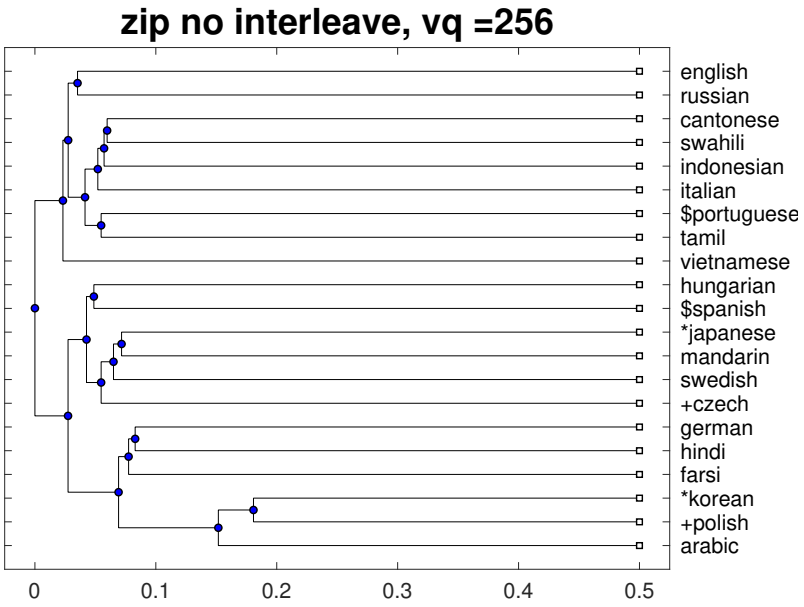


(a) without interleave

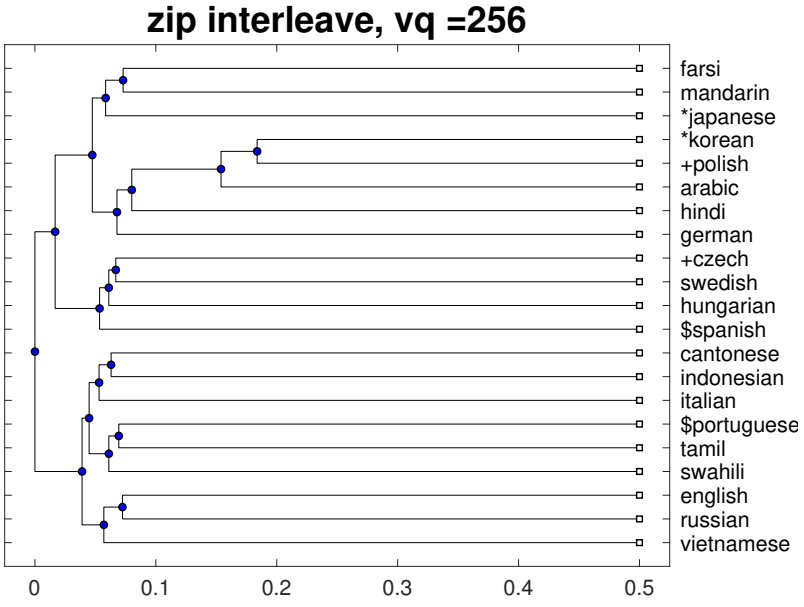


(b) with interleave

Figure 4.40: The 21 UNDHR audio languages distances are computed by ppm and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 256. Figure 4.40(a) shows the non-interleaved result and Figure 4.40(b) shows the interleaved result.

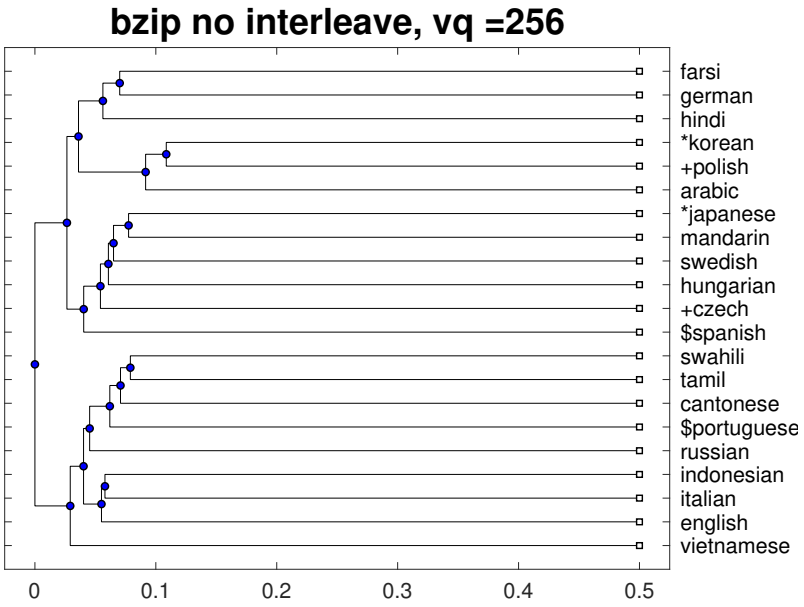


(a) without interleave

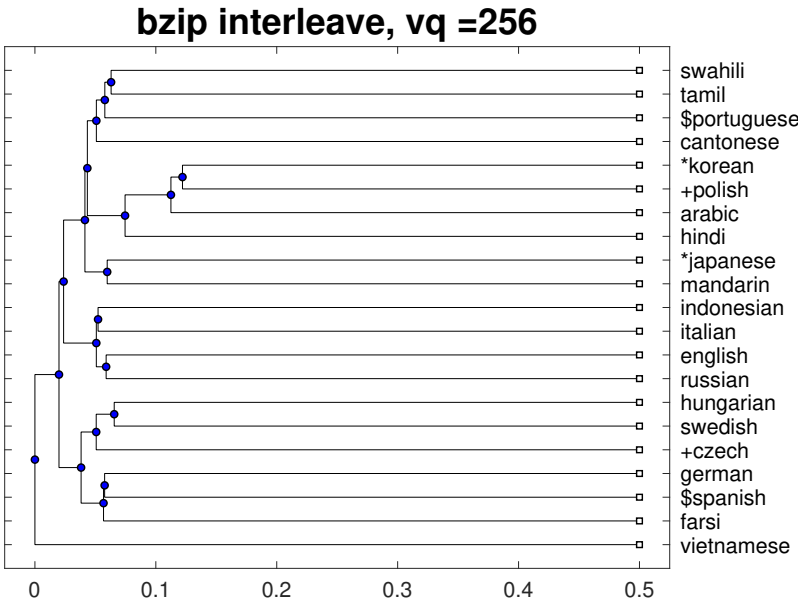


(b) with interleave

Figure 4.41: The 21 UNDHR audio languages distances are computed by zip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 256. Figure 4.41(a) shows the non-interleaved result and Figure 4.41(b) shows the interleaved result.

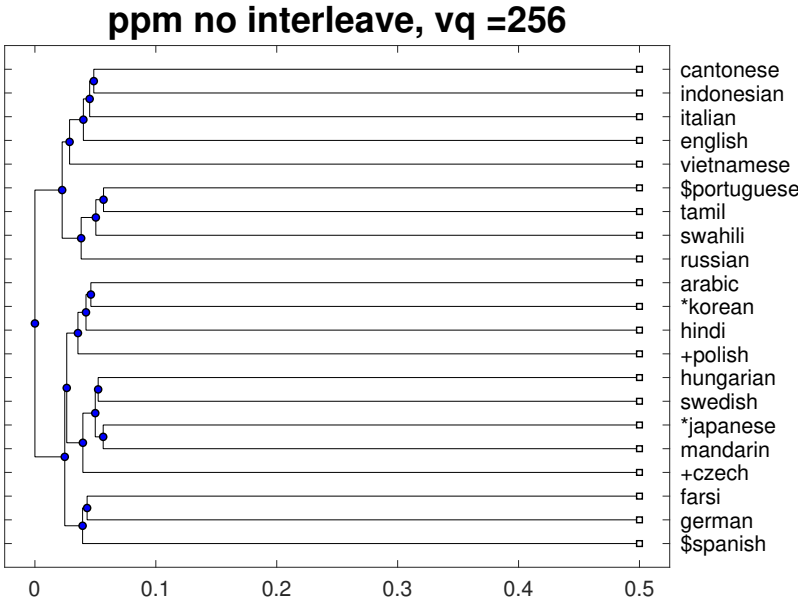


(a) without interleave

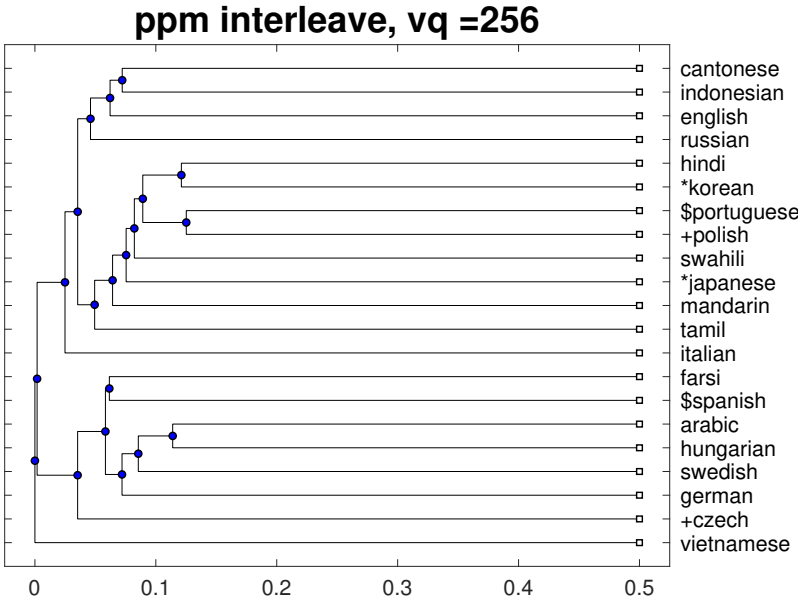


(b) with interleave

Figure 4.42: The 21 UNDHR audio languages distances are computed by bzip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 4.18(a) shows the non-interleaved result and Figure 4.18(b) shows the interleaved result.



(a) without interleave



(b) with interleave

Figure 4.43: The 21 UNDHR audio languages distances are computed by ppm and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 4.19(a) shows the non-interleaved result and Figure 4.19(b) shows the interleaved result.

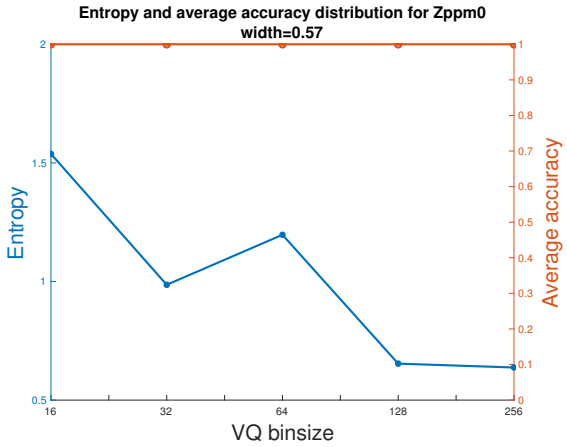
4.4.7 Conclusion

Figure 4.44 compares the entropy and the accuracy of zipping methods. The error bars are not displayed because the accuracies are all 100%. The reason that ppm shows a lower entropy in the 32 VQ bins than in the 16 VQ bins is it can predict the "escape" characters. As the compressor use the fixed length of blocksize to control the length of the strings for zipping, the "escape" characters mean the characters which are not shown in the first blocksize but shown in the next one. If the compressor does not know the characters, it has to use more space in the compressed string to describe them. As we know the 16 VQ bins case contains a lot of long repeated characters, the character set in two blocksize might be very different - the amount of the "escape" characters might be high. The 32 VQ bins case has more variant in characters thus it has shorter long repeat character. It means the number of the "escape" characters is less than the 16 VQ bins and the strings become predictable. From the 32 VQ bins to the 64 VQ bins, the entropy increases is because the size of the character set increases. As the increasing of the VQ binsize, the differences in and between the languages become large. It makes the distances between the languages large and worse the compressibility - a higher entropy of the language distance distribution. The ppm interleaved results perform better because the interleaving solves the problem that the non-interleaving faces - the "escape" characters.

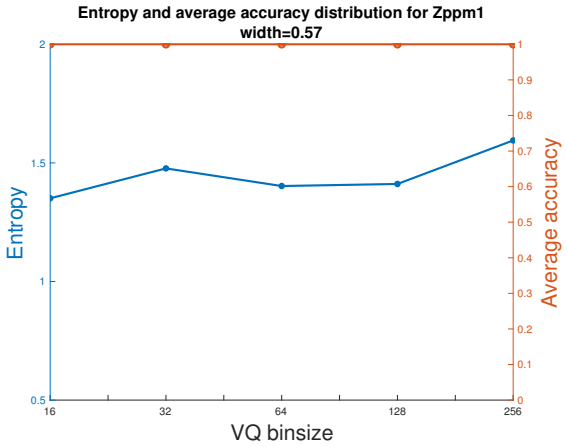
The entropy distribution of the ppm shows that the compressibility of the ppm is stable if it can predict all characters at the very first time. For the zip method, the LZ77 simply compresses the duplicate characters which are neighbours. That means the zip is not good at dealing with the irregular string, for example, the interleaved string which does not contains so many repeated and connected characters. That is why the entropy of the zip with interleaved method is larger than the zip with non-interleaved. For the non-interleaved result, the entropy goes high because the 256 VQ bins case has more variant of characters and fewer long repeated characters. The interleaved results show more character variations inside the buffer string and the entropies increases along with the number of VQ bins. Since bzip is over-sensitive to

long repeated characters, the 16 VQ bins get a high entropy than the 64 VQ bins in the non-interleaving result. For the 32 VQ bins, as it still contains a lot of repeated characters inside the language and more character variation between languages, it shows a higher entropy than the 16 VQ bins. The 64 VQ bins show the impact of larger characters and less repeated characters. The 32 VQ bins shown in interleaved result also describe the impact of repeated characters inside the language and the differentiation of the characters between the languages. The entropy decreased in the 128 and the 256 VQ bins in the interleaved result is because of the loss of information as some languages might share more characters. The interleaved characters will be sorted by BWT and transformed into long repeated characters and re-calculated by the run-length encoding. Thus, the Bzip model performs better compressibility and views these languages as similar to each other.

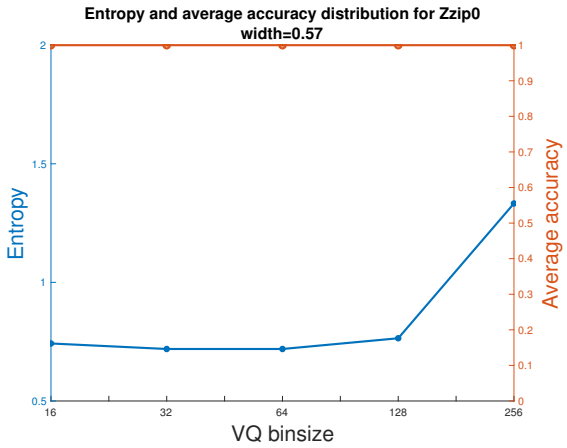
According to the zipping results, we can see the audio features are different from the text since it contains long repeat Unicode character strings which are explained in Section 4.4.2. Not like n-gram model, zipping cannot identify the internal relationships of those long strings and simply ignore them for a better compression performance. Also, the blocksize of zipping limits the ability to predict the unknown characters. Thus, we can say that the differences between the ALID languages are not distinctive. However, it still can build the language tree based on the distance matrix. So we are going to compare the highest entropy - the ppm tree with the linguistic tree and the TLID tree in Chapter 6.



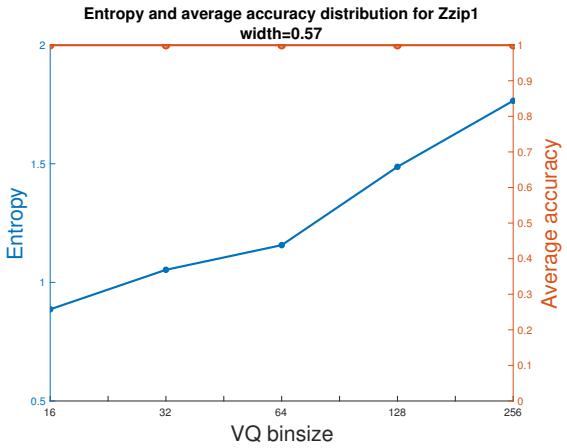
(a) Accuracy and entropy distribution for Zppm0



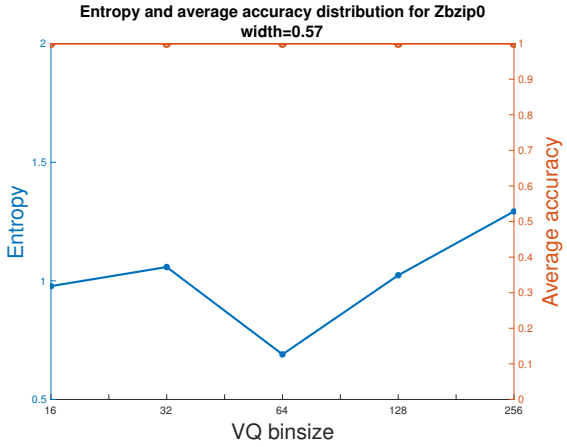
(b) Accuracy and entropy distribution for Zppm1



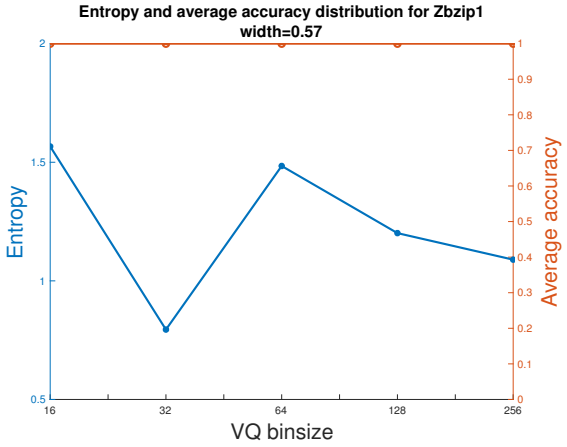
(c) Accuracy and entropy distribution for Zzip0



(d) Accuracy and entropy distribution for Zzip1



(e) Accuracy and entropy distribution for Zbzip0



(f) Accuracy and entropy distribution for Zbzip1

Figure 4.44: Accuracy and entropy distribution for zip, ppm and bzip with interleaved and non-interleaved data. The x-axis is the number of VQ bins from 16 to 256. The left y-axis is the entropy value and the right y-axis is the accuracy value.

4.5 CK distance using MPEG

4.5.1 Introduction

In this section, we will discuss the language distance, proposed by Campana and Keogh [2010]: the CK-distance. The technical details of the CK-distance are described in Section 4.5.2, so just a brief explanation is given here. The main question discussed by Campana and Keogh [2010] is whether it is possible to find similarities by using video compressors, such as MPEG. Campana and Keogh [2010] use CK distance for texture analysis based on five different features in the psychology of perception: coarseness, contrast, directionality, line-likeness and roughness, which is proposed by Tamura et al. [1978]. They claim CK-distance performs high recognition accuracy in species classification and breast cancer identification. The features for MPEG can be global scalars and global vectors such as energy, entropy, and wavelet coefficients. Like bzip, zip and ppm, the image compressor for CK-distance mainly works on the similarities of two images. Image compressors reduce the size of images by creating a "video" and compare to the size of original images. Campana and Keogh [2010] claims two images are similar if the compressor produces a smaller size of the file which can be viewed as a significant similarity.

Base on Campana and Keogh [2010]'s idea of image classification, we come up with an idea that if it is possible to apply the same method for audio language identification. And in fact, Hao et al. [2012] proposed that CK-distance also works for insect sounds classification by using MPEG. In that case, we can create spectrograms for the waveform. By using the video compressor, we wonder if CK-distance can find the similarities between the languages.

4.5.2 Methods

Figure 4.45 explains the procedure of CK distance. The procedure is similar to zip, bzip and ppm but we use waveform instead of MFCCs for generating the spectro-

grams to show the spectral characteristics varies during the time period. We also introduce 10-fold cross validation and use the UNDHR 21 language corpus, where the sampling frequency rate is $8kHz$. The waveforms are chunked into subsignal by 0.5 seconds for each spectrum. The waveforms are converted into spectrograms $\boldsymbol{S}_i, i \in (1...n)$ by using the short-time Fourier transform, which is a sequence of short overlapping DFTs, see Equation 4.7:

$$\boldsymbol{S}(m,i) \triangleq |DFTy(k)x_m k|, 0 \leq m < 2M-1, 0 \leq i < L \tag{4.7}$$

$\boldsymbol{S}(m,i)$ means the spectrogram matrix of signal sequence $\boldsymbol{x}(k)$. $2M-1$ means the number of subsignals and the length of $x(k)$ is L. $w(k)$ is the windowing function which in our case is the Hamming window for DFT (Discrete Fourier Transform). The purpose of using the window function $w(k)$ is to reduce frequency domain leakage [Schilling and Harris, 2012]. The concepts of the Hamming window and DFT are described in Section 4.2.

After concatenating $\boldsymbol{S}_i$ and $\boldsymbol{S}_j$, $i,j \in (1...n)$, the combined spectrograms are compressed by the MPEG compressor. The sizes of the compressed files are labelled as $l(m_{ij}), i,j \in (1...n)$. The CK-distance we used in this section is in equation 4.8.

$$d_{ab} = \frac{m(a|b) + m(b|a)}{m(a|a) + m(b|b)} - 1 \tag{4.8}$$

The distance d_{ab} means the difference between image a and b and $m(a|b)$ stands for the size of compressed images a and b but image b is attached after image a.

The generated CK-distances are regarded as the distances, or difference, between languages. To look into the difference between languages and ease to compare with n-gram, bzip, zip and ppm results, we use a colour map, a phylogenetic like tree and a histogram distribution to explain the CK-distance result.

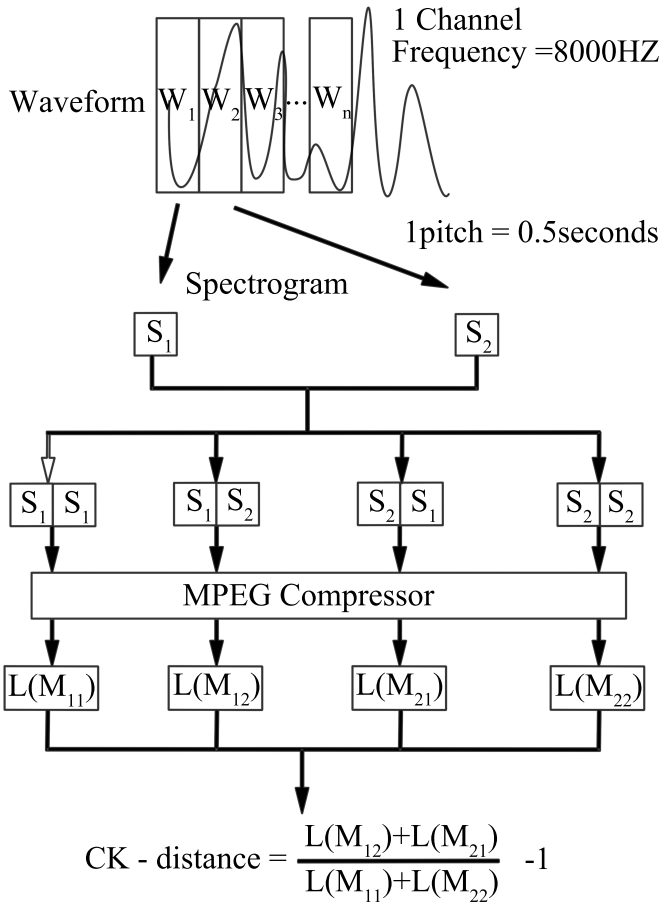


Figure 4.45: CK-distance procedure. This model introduces MFCC features to generate a spectrogram and calculate the CK-distance between spectrogram images. The UNDHR 21 languages datasets are used for both training and testing.

4.5.3 CK-distance results

This section describes the language distances via CK-distance. The language distances are represented by a colour map, a phylogenetic like tree and a histogram distribution. The colour map displays the colour density of distance. Figure 4.46 shows the colour map of the languages distances which displays the data as an image that uses the full range of RGB colors. The description of phylogenetic tree is in Section 3.2.2 and the description of histogram distribution is in Section 3.2.1.2. Figure 4.47 shows the phylogenetic like dendrogram of language distances.

Based on the linguistic language tree in Section 2.7, we can define three language

subsets - Spanish and Portuguese, Korean and Japanese, Czech and Polish. In the colour maps, we denote the Spanish and Portuguese by pink, Korean and Japanese by blue and Czech and Polish by red. In the dendrogram, we denote Spanish and Portuguese as symbol "$", Korean and Japanese as symbol "$*$" and Czech and Polish as symbol "+".

Figure 4.48 shows the histogram distribution for MPEG. The entropy value of the histogram distribution is 0.71. The bin width of the histogram is also calculated the same as previous chapters that bin width is w/σ, which is previously discussed in Section 3.2.1.2. It means the distances shown in the diagram are $distance/\sigma$.

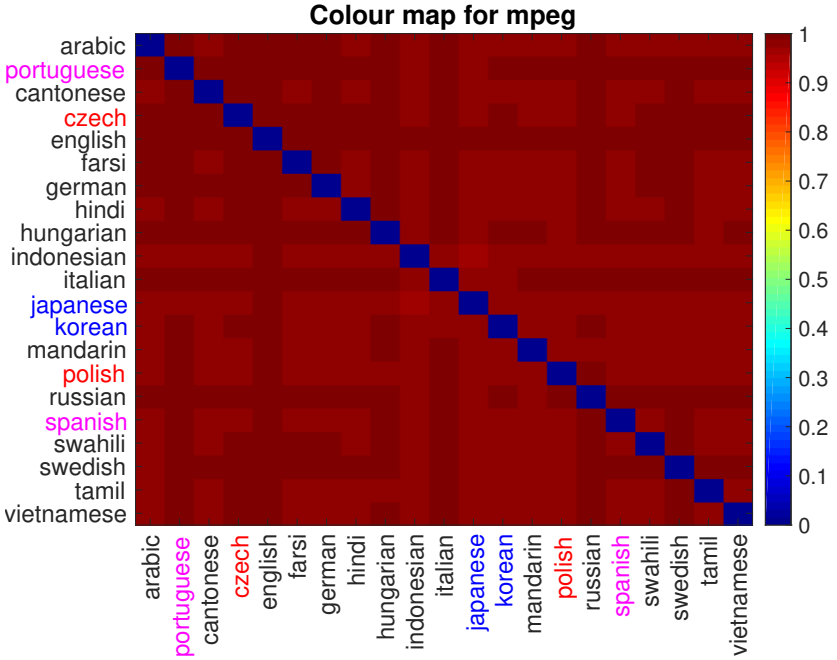


Figure 4.46: The 21 UNDHR audio languages CK-distances calculates the size of compressed images using MPEG. The distances shown in the colour map are $distance/\sigma$ and are normalized into $[0, 1]$.

4.5.4 Conclusion

This section discusses whether we can use CK-distance as a language distance. The CK-distance requires the use of MPEG to compress the images and to estimate the size of the compressed image files $L(M_i), i \in (1...n)$. The CK-distance measures

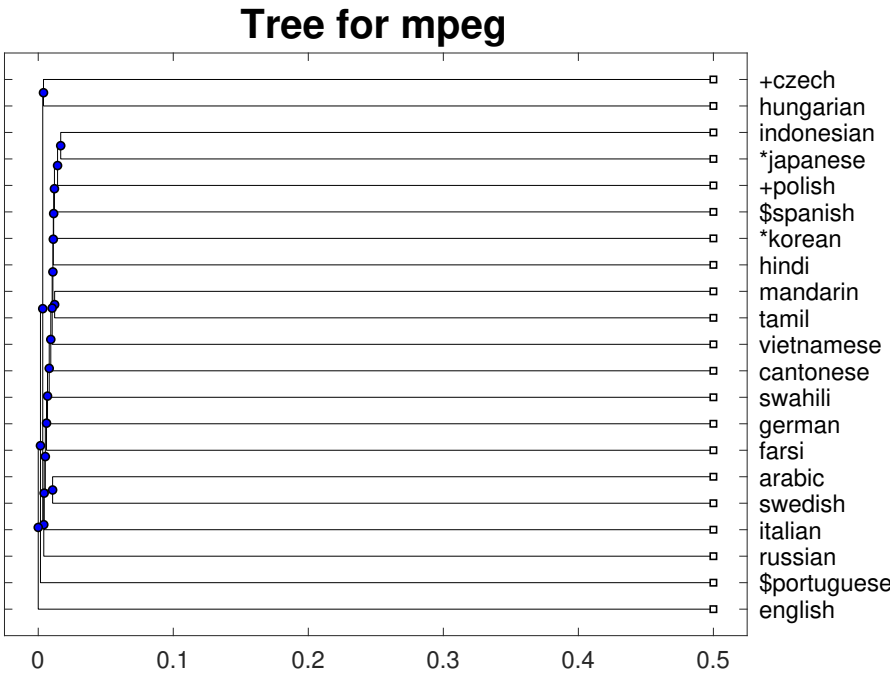


Figure 4.47: The 21 UNDHR audio languages CK-distances calculates the size of compressed images using MPEG and displayed by the dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The lengths of the branches between the points correspond to the distances between the languages.

differences in $L(M_i)$ and we describe the CK-distance matrix using colour maps and tree distance structures.

We can see that the MPEG gets 100% in language classification and can find the spectrograms that are the same as itself. Thus, MPEG works for identifying different languages and can tell which language it is under the condition that the language is collected in the database. However, we can see that the distances between the different languages shown in Figures 4.46 and 4.47 slightly differ. Figure 4.48 shows the histogram of language distances distribution. The entropy of histogram distribution of CK-distance is 0.71, which is much lower than Cavnar and Trenkle [1994]'s n-gram model (Section 4.3) and other compressor. We can find this is another "all-or-nothing" classifier. Considering the time period of the MPEG compression, it is also not suitable for fast language identification in the emergency case. Thus, the MPEG is not fit for our requirement.

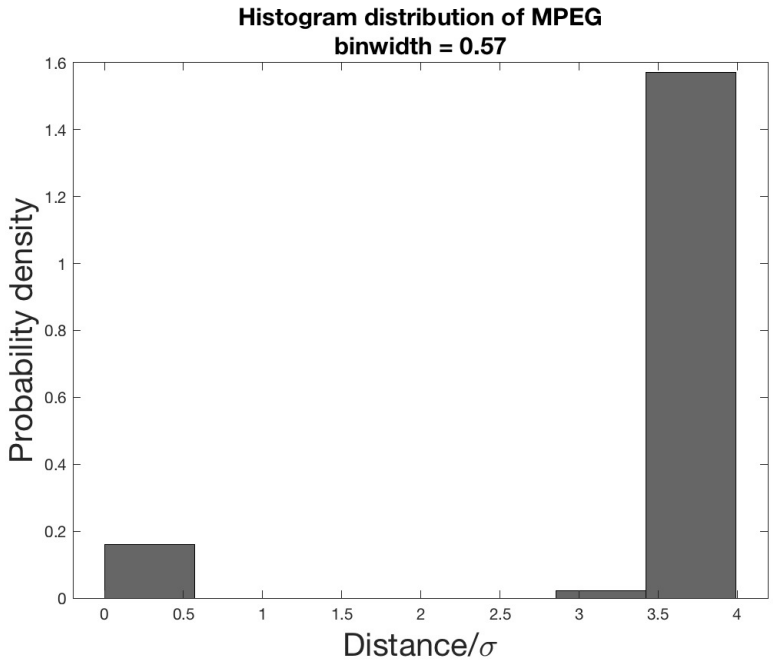


Figure 4.48: The 21 UNDHR audio languages CK-distances calculate the size of compressed images using MPEG. The histogram shows the distance distribution of languages.

4.6 Conclusion

In this chapter, we examine three methods for ALID. We might expect that, for audio signals, the best performance would be obtained with methods that work for the text domain. In fact, we find the evidence that the benchmark method in text language classification, the Cavnar and Trenkle [1994]'s n-gram method, also works well in audio for language identification. Of course one has to find an acceptable audio representation (we use MFCCs) and quantise it, but otherwise it works. For the language relationships, we can see the trees which are built by the n-gram and the zipping are not similar to the linguistic language tree. As we explained in Section 4.3.6, the linguistic tree is not built by all linguistic rules and the linguists view some unique rules more important than others. However, we still can find some Indo-European languages are close to each other. We are going to talk about the similarity and differences of the ALID languages trees, the TLID trees, and the linguistic language trees in Chapter 6.

A further advantage of the n-gram techniques arises from its independence of phonemes. As the vector quantised MFCCs contains long repeated characters, it is different from text languages since text are already vector quantised by Unicode and has a huge variety in character set. Both zipping and CK-distance by MPEG do not concern the internal relationships between those characters, which make the entropy of zipping and CK-distance lower than n-gram. As the CK-distance is another "all-or-nothing" methods, and also the text is vector quantised by Unicode and not suitable for image compression, we are not going to further investigate the MPEG results in TLID and VLID.

Chapter 5

VLID (Video Language IDentification) results

5.1 Introduction

This chapter is going to apply Cavnar and Trenkle [1994]'s n-gram model and zipping methods to the VLID system. As previous chapters have been shown that both n-gram model and zipping methods have high recognition accuracies and entropies, by applying these methods to video data, we expect the languages can be identified and also show the distance relationships between each other. What we expect is, in VLID, the Cavnar and Trenkle [1994]'s n-gram model still gets a higher accuracy and entropy than zipping methods.

The database we used in this chapter is the Universal Declaration of Human Rights (UNDHR) dataset, which was recorded by Jacob Newman. The database records English, Arabic and Mandarin speakers who read the UNDHR. A detailed description of the video datasets is presented in Section 2.6.

5.2 Cavnar and Trenkle's n-gram model

This section will apply Cavnar and Trenkle [1994]'s n-gram model to the VLID system. Like Cavnar and Trenkle [1994]'s n-gram model work on ALID, in VLID, the dataset is not sequenced data, such as strings. The extracted AAMs are vectors that contain shape and appearance features, so the n-gram model cannot be used directly for calculating the frequency of AAM features.

Figure 5.1 details the procedure of Cavnar and Trenkle [1994]'s n-gram model working on video AAM features. First, the system vector-quantises the AAMs into 16, 32, 64, 128, 256 bins. Each bin is represented by a character so the sequence of bins is then written as a sequence of characters into text files.

We use 10-fold cross-validation in the experiment. The vector-quantised AAMs of each language were chunked into 10 folds, with 9 folds for training the n-gram frequency models and 1 fold for testing these models. We compare the distance between the training frequency model and the testing frequency model and used the difference in rank as the n-gram distance. If the n-grams do not exist in the training or testing frequency vector, we charge a maximum penalty of 400 as their distance. As with ALID, we examine the effect of the penalty parameter.

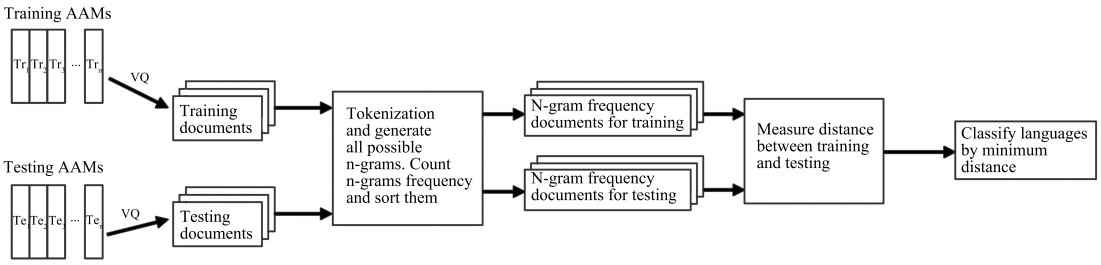


Figure 5.1: Cavnar and Trenkle [1994]'s n-gram frequency model for UNDHR video dataset provided by Newman [2011].

5.2.0.1 AAM (Active Appearance Models)

The active appearance model (AAM) is an extension of the active shape model (ASM) and is used to interpret images as a set of parameters without the loss of important information [Cootes et al., 2001b]. The ASM concentrates on modelling the shape of images while the AAM tries also to generate and interpret the appearance represented by texture and colours. Both AAMs and ASMs can be used for high-level interpretation of images, as well as image reconstruction.

To enhance the flexibility of models, AAMs and ASMs generate a 'shape-free' appearance by pre-defining all images in the training set to have the same shape. In this project, we use labels, or 'landmarks', to constrain the boundary of the shape and the landmarks can be adapted in all images.

To generate a robust and flexible ASM model, the training set $P = \{p_1, p_2, p_3, ..., p_n, n \in N\}$ are generated from images set N by

$$p_n = \overline{P} + Eb, \quad n \in N \tag{5.1}$$

where p_n is the training examples and p_n could be the shape or colour. $\overline{P}$ is the means of the training examples. E is the matrix of orthogonal modes of the training set and b is the weight parameters of eigenvectors E used for controlling shape and texture. The parameters in b are initialised to 0 and only change one at a time with ± 3 standard deviations from the means $\overline{P}$.

In shape part, the training set $P(n)$ is generated by an in-plane rotation σ, a scaling S and a translation $T = (T_x, T_y)$. The rotated scaling $(S_{\sigma x}, S_{\sigma y})$ is determined by rotation and scaling, which is $S_{rx} = (S\cos(\sigma - 1))$ and $S_{ry} = S\sin\sigma$. Assuming the transpose matrix $V = (S_{rx}, S_{ry}, T_x, T_y) \approx 0$ for identity transformation, then the corresponding shape set $P_{V+\delta V}(n)$ is close to $P_V(P_{\delta V}(n))$ [Cootes et al., 2001b].

In appearance part, the training set P is generated by a scaling S and an offset F to the intensities, which means $P(n) = (S + 1)P(n) + F1$, 1 stands for a unit vector. Assuming $V = \{S, F\}$ is the vector of transformation and $S \approx 0$ and $F \approx 0$,

then the appearance set $P_{V+\delta V}(n) \approx P_V(P_{\delta V}(n))$[Cootes et al., 2001b].

Since the shape and appearance are trained separately, AAM applies PCA to overcome the problem of combining shape and appearance parameters and reducing the dimension. To reduce the unbalance significance between shape and appearance, Cootes et al. [2001b] mentions that it is necessary to normalise both the shape and appearance vectors.

5.2.1 Language distance results with 16 bins

In this section, we examine the results of Cavnar and Trenkle [1994]'s model applied to video. As we previously said, the facial features are all converted into AAMs and the AAMs are all converted into symbols (Unicode characters). By using Cavnar and Trenkle [1994]'s n-gram model, the differences of n-grams frequencies can determine the differences between languages. We add a penalty to describe the impact of the n-gram which is not been seen in the other languages.

Table 5.1 to 5.5 show the entropy and accuracy of each penalty. We use the 16, 32, 64, 128 and 256 VQ bins in this experiment. The accuracy and its standard error are computed as the mean and standard error of the ten test accuracies from each folder using an n-gram classifier trained on the training data in each fold. Each fold also produces a distance matrix which are the distances between the test languages in that fold as measured by the n-gram method trained on each training fold. The mean of these distances is summarised by the entropy. We can find the accuracy of identification language is always low by using n-gram.

Table 5.1 shows the accuracy and entropy of Cavnar and Trenkle [1994]'s n-gram model with 16 VQ bins. Figure 5.2 compares the accuracies and entropies, the accuracy has error bars ± 2 standard error. Considering the highest accuracy and entropy, we find the best performance is the tri-gram (Figure 5.2(c)), whose penalty is 5. For uni-gram with 16 VQ bins, there are only 16 n-grams and these three languages share the same character set. So the accuracy is the same for all penalties.

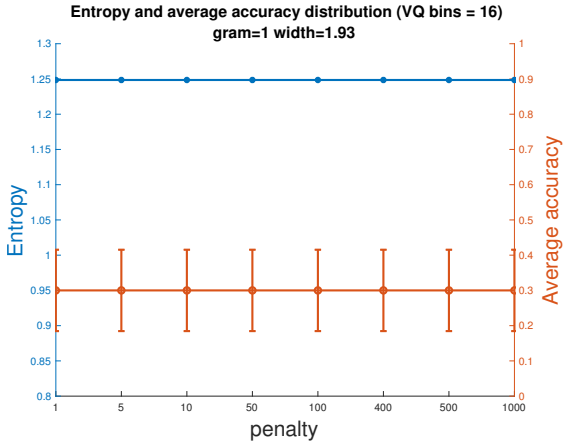
Table 5.1: Entropy(top) and accuracy(bottom) values with histogram binwidth = 1.93, vq bin size = 16.

	Penalty value							
	1	5	10	50	100	400	500	1000
	Entropy value							
Gram=1	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25
Gram=2	0.97	0.97	0.97	0.97	0.97	0.97	0.97	1.01
Gram=3	0.97	1.14	0.97	1.25	0.97	0.97	0.97	0.97
Gram=4	1.25	0.97	0.83	0.97	0.97	0.83	1.25	0.83
Gram=5	1.25	0.97	1.25	0.97	1.14	0.83	1.25	0.83
	Accuracy value							
Gram=1	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30
Gram=2	0.33	0.33	0.33	0.33	0.30	0.30	0.30	0.30
Gram=3	0.37	0.37	0.33	0.27	0.27	0.30	0.33	0.30
Gram=4	0.20	0.20	0.23	0.17	0.20	0.20	0.20	0.23
Gram=5	0.37	0.37	0.33	0.20	0.20	0.27	0.27	0.27
	Standard error							
Gram=1	0.06	0.06	0.06	0.06	0.06	0.06	0.06	0.06
Gram=2	0.03	0.03	0.03	0.03	0.06	0.06	0.06	0.06
Gram=3	0.07	0.07	0.03	0.03	0.03	0.06	0.07	0.12
Gram=4	0.15	0.15	0.13	0.07	0.06	0.10	0.10	0.07
Gram=5	0.09	0.09	0.09	0.06	0.00	0.03	0.03	0.03

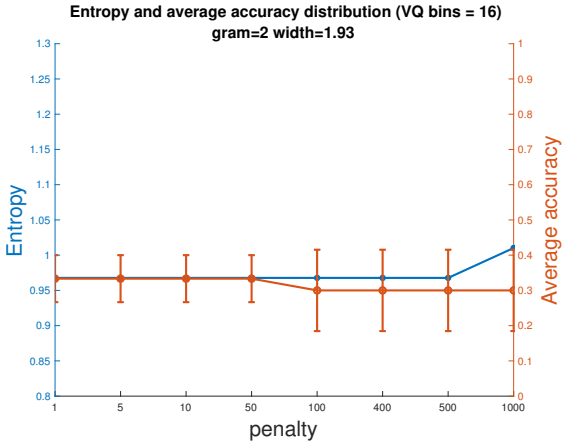
However, unlike ALID results, there are only three languages in video dataset. The histogram bin width is based on Equation 3.5 which means it is inversely related to the number of distance results. For this reason, the binwidth value of histogram is larger than the TLID and the ALID. A large histogram binwidth means the language distances can only be binned into a small number of bins - the worst case is 2 bins. This indicates a very non-smooth histogram, which means low entropy. Additionally, the long repeated characters in the TLID are longer than the ALID. So, as we use the 10-fold cross validation, after the data are split into 10 parts, it is possible that the strings for training are different from the strings for testing. This causes the distances of languages to vary as the rank of the n-gram occurrences in the training and testing are different. AS the entropy is the average of the language distances, it is not surprising that the entropy performs random in the plot. For accuracy, since one is guessing randomly between the three video languages then the accuracy is $\frac{1}{3}$,

we conclude that the n-gram does not work on VLID for 16 VQ bins.

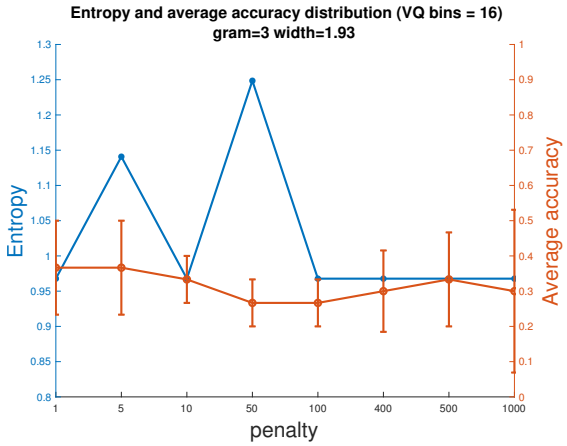
Figure 5.3 visualizes the tri-gram, 5 penalty result in 16 VQ bins. Figure 5.3(a) shows the colour map of languages and Figure 5.3(b) shows the dendrogram which is built based on $d = distance/\sigma$ where d is normalized into $[0, 1]$. The dendrogram is built based on complete-linkage clustering (explained in Section 3.2.2). However, Figure 5.3(a) shows that English is more closer to Arabic rather than itself - a bad language identification. Since there is no clue in linguistic language tree to present the relationships between English, Mandarin and Arabic, we compare the distances with ALID result with 16 bins. We can find the distances between Arabic and English in Figure 5.3(b) are closer than Arabic and Mandarin while Arabic is more closer to Mandarin in ALID with the same VQ bins (See Figure 4.5(b)). In that case, we think the 16 VQ bins case still performs poorly in VLID.



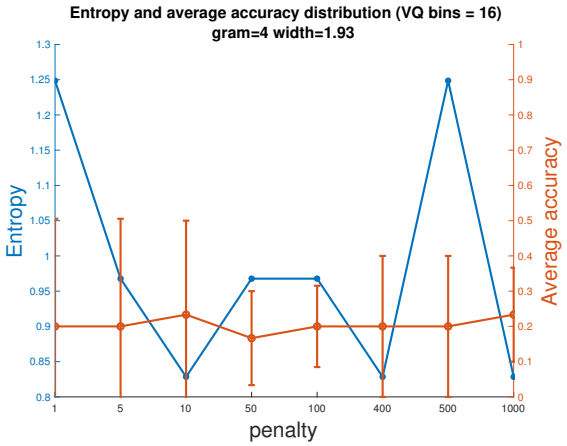
(a) Accuracy and entropy distribution for gram = 1



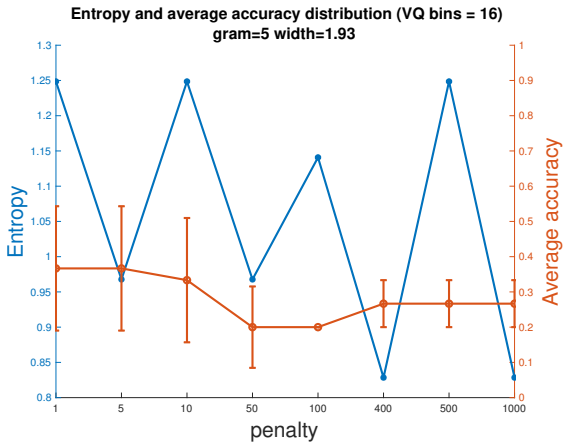
(b) Accuracy and entropy distribution for gram = 2



(c) Accuracy and entropy distribution for gram = 3

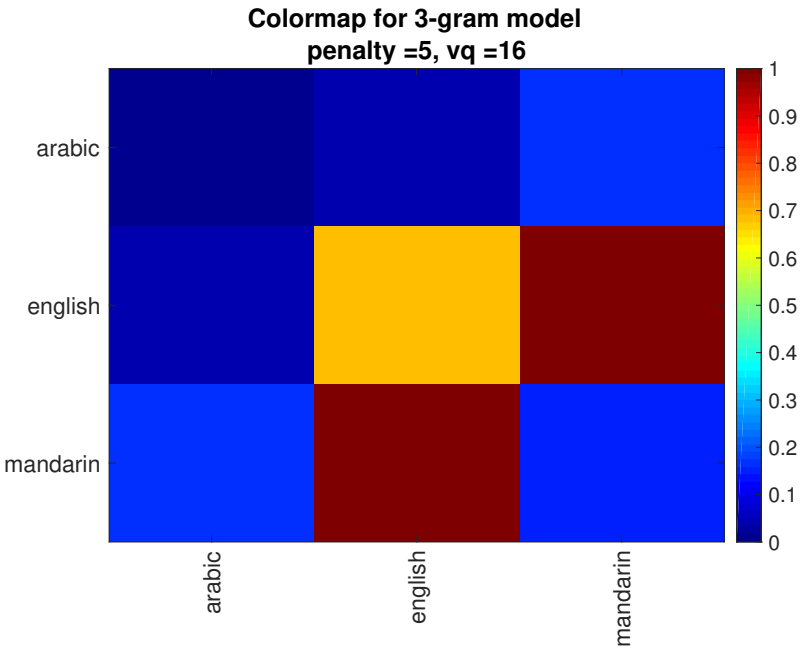


(d) Accuracy and entropy distribution for gram = 4

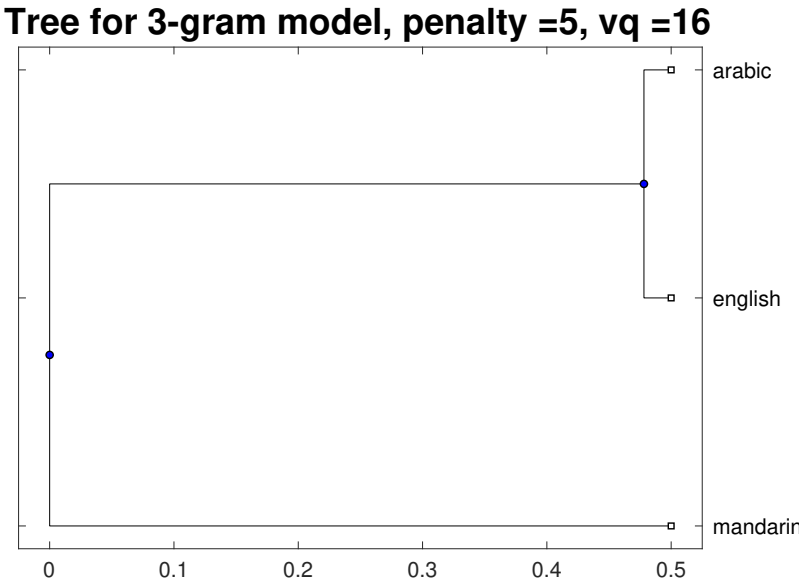


(e) Accuracy and entropy distribution for gram = 5

Figure 5.2: Accuracy and entropy distribution for n-grams. VQ bin size is 16. The x-axis is the penalty value. The left y-axis is the entropy value and the right y-axis is the accuracy value.



(a) Colormap of tri-gram



(b) dendrogram of tri-gram

Figure 5.3: The video language distances results of tri-gram for English, Mandarin and Arabic. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 5 and the VQ bins is 16. Figure 5.3(a) shows the colour map of the language distance variations and Figure 5.3(b) shows the language tree which is built by the distances. The colour variation in Figure 5.3(a) shows the pairwise distances between languages.

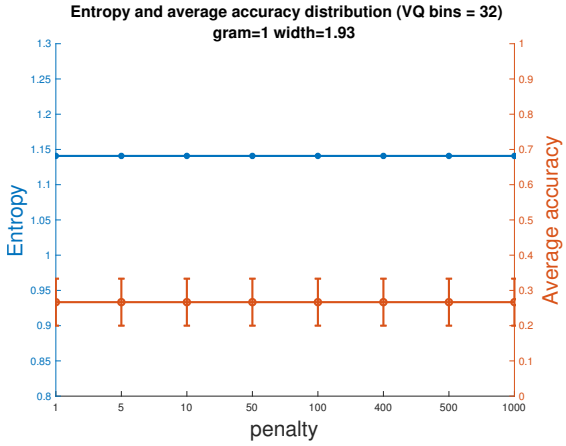
5.2.2 Language distance results with 32 bins

Table 5.2: Entropy(top) and accuracy(bottom) values with histogram binwidth = 1.93, vq bin size = 32.

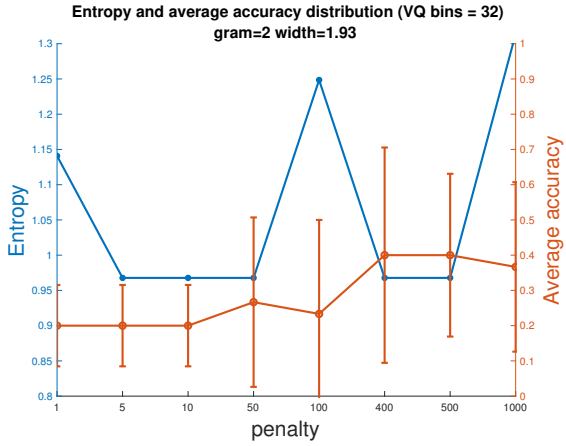
	Penalty value							
	1	5	10	50	100	400	500	1000
	Entropy value							
Gram=1	1.14	1.14	1.14	1.14	1.14	1.14	1.14	1.14
Gram=2	1.14	0.97	0.97	0.97	1.25	0.97	0.97	1.31
Gram=3	0.97	0.97	1.01	0.83	0.83	0.83	1.25	0.83
Gram=4	0.97	0.97	0.97	0.97	1.25	0.83	1.14	0.83
Gram=5	0.97	1.25	1.01	1.14	1.25	1.25	0.83	1.25
	Accuracy value							
Gram=1	0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27
Gram=2	0.20	0.20	0.20	0.27	0.23	0.40	0.40	0.37
Gram=3	0.40	0.40	0.40	0.37	0.27	0.20	0.20	0.23
Gram=4	0.43	0.43	0.43	0.30	0.27	0.27	0.30	0.30
Gram=5	0.40	0.40	0.37	0.30	0.27	0.20	0.17	0.20
	Standard error							
Gram=1	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
Gram=2	0.06	0.06	0.06	0.12	0.13	0.15	0.12	0.12
Gram=3	0.06	0.06	0.06	0.07	0.09	0.06	0.06	0.07
Gram=4	0.07	0.12	0.12	0.12	0.03	0.03	0.06	0.06
Gram=5	0.10	0.10	0.12	0.12	0.07	0.06	0.07	0.06

Table 5.2 shows the accuracy and entropy of Cavnar and Trenkle [1994]'s n-gram model with 32 VQ bins. Figure 5.4 compares the accuracies and entropies, the accuracy has error bars ±2 standard error. Considering the highest accuracy and entropy, we find the best performance is the tri-gram (Figure 5.4(c)), whose penalty is 10. And also we conclude that the n-gram does not work on VLID for 32 VQ bins in most penalty cases since they are guessing randomly between the video languages. 32 VQ bins case also has the same problem as 16 VQ bins that, a large histogram binwidth means the language distances can only be binned into a small number of bins which indicates a very spiky histogram which means low entropy. We find the accuracy decreases with the increasing of penalty for gram 3, 4 and 5, which proves that the bottom-ranked n-grams contain useless information and confuses the classifier.

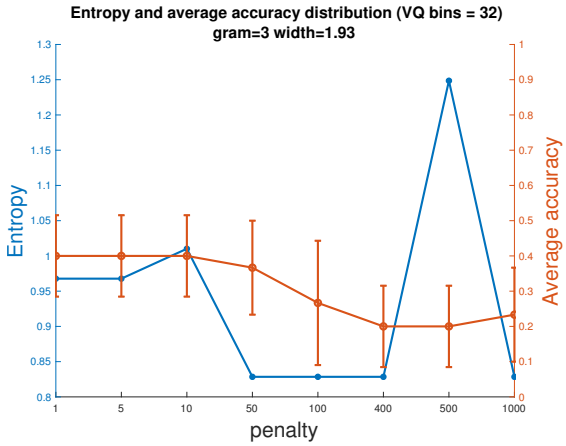
Figure 5.5 visualizes the tri-gram, 10 penalty result in 32 VQ bins. Figure 5.5(a) shows the colour map of languages and Figure 5.5(b) shows the dendrogram which is built based on $d = distance/\sigma$ where d is normalized into $[0, 1]$. The dendrogram is built based on complete-linkage clustering (explained in Section 3.2.2). However, the video result shown in Figure 5.5(a) shows that English is closer to Mandarin rather than itself which is also a bad language identification. Since there is no clue in linguistic language tree to present the relationships between English, Mandarin and Arabic, we compare the distances with the ALID result with 32 bins. We can find the distances between Mandarin and English in Figure 5.5(b) are closer than Mandarin and Arabic while Arabic is more closer to English in the ALID with the same VQ bins (See Figure 4.7(b)). And compared with the 16 VQ bins case, the tree is the opposite conclusion of 16 VQ bins. Considering the low accuracy of the results, the variations of the language distances distributions are random. So, the average distance of the 10-fold cross validation results is unreliable. In that case, we think the 32 VQ bins case still performs poorly in the VLID.



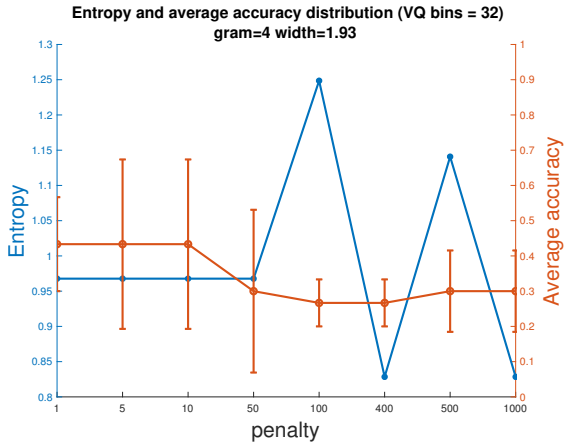
(a) Accuracy and entropy distribution for gram = 1



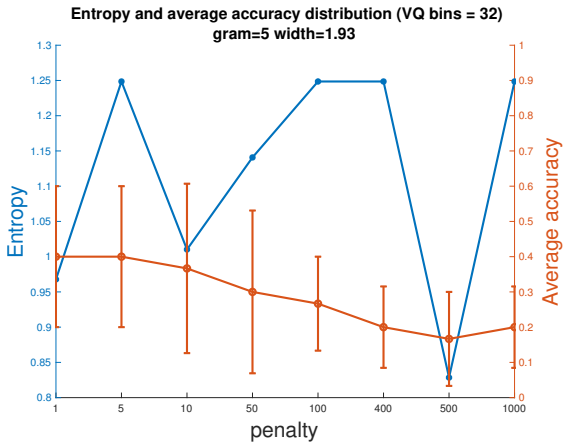
(b) Accuracy and entropy distribution for gram = 2



(c) Accuracy and entropy distribution for gram = 3

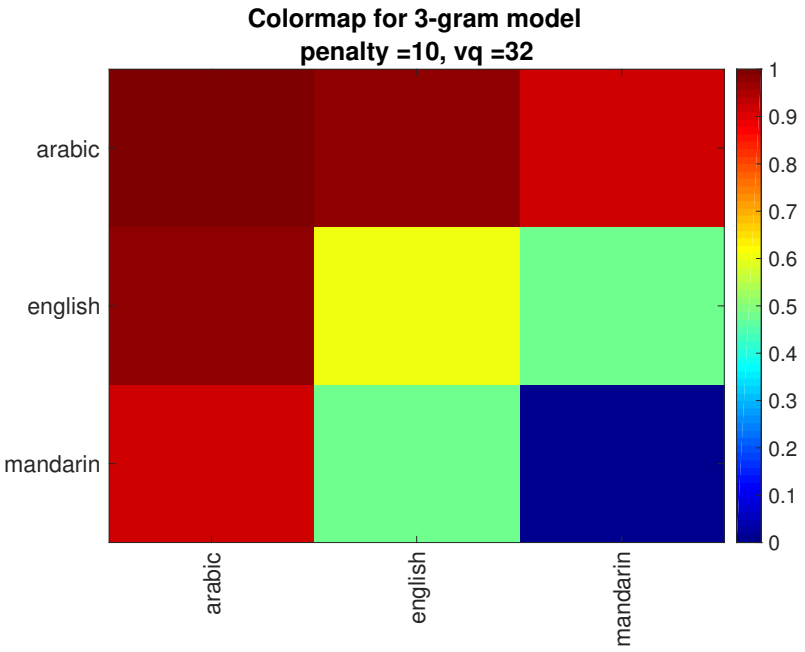


(d) Accuracy and entropy distribution for gram = 4

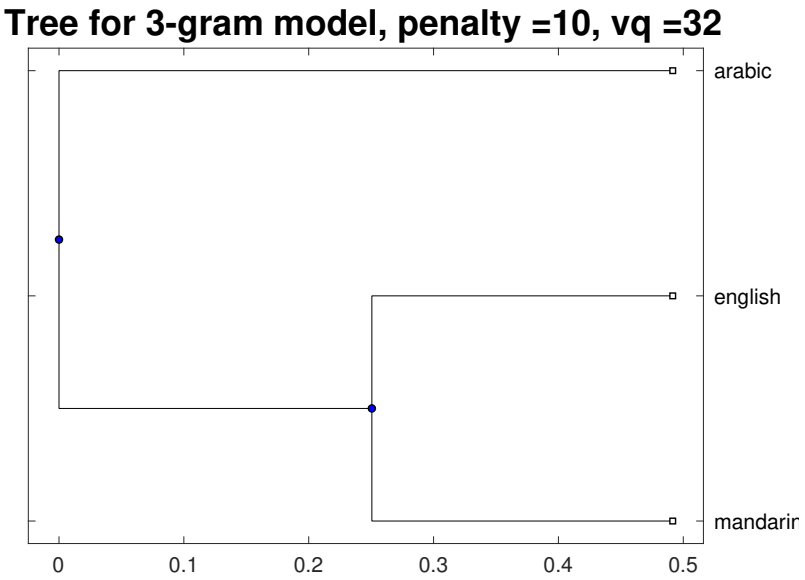


(e) Accuracy and entropy distribution for gram = 5

Figure 5.4: Accuracy and entropy distribution for n-grams. VQ bin size is 32. The x-axis is the penalty value. The left y-axis is the entropy value and the right y-axis is the accuracy value.



(a) Colormap of tri-gram



(b) dendrogram of tri-gram

Figure 5.5: The video language distances results of tri-gram for English, Mandarin and Arabic. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 10 and the VQ bins is 32. Figure 5.5(a) shows the colour map of the language distance variations and Figure 5.5(b) shows the language tree which is built by the distances. The colour variation in Figure 5.5(a) shows the pairwise distances between languages.

5.2.3 Language distance results with 64 bins

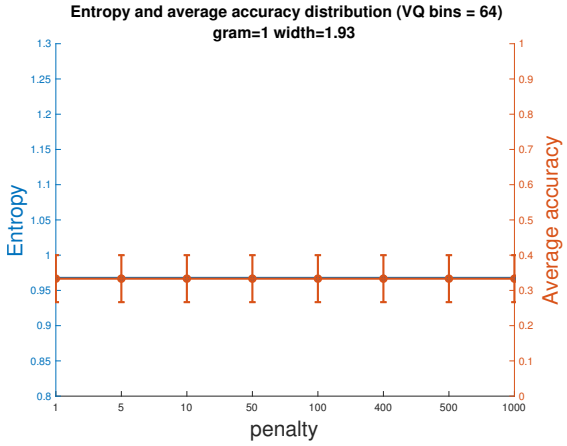
Table 5.3: Entropy(top) and accuracy(bottom) values with histogram binwidth = 1.93, vq bin size = 64.

	Penalty value							
	1	5	10	50	100	400	500	1000
	Entropy value							
Gram=1	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97
Gram=2	1.25	1.14	1.14	0.97	1.25	0.97	0.97	0.97
Gram=3	0.97	1.14	1.14	1.14	1.14	1.25	0.83	1.14
Gram=4	1.25	0.97	1.14	0.83	1.14	0.97	0.83	0.83
Gram=5	1.14	1.14	1.14	1.01	1.25	1.01	1.25	0.97
	Accuracy value							
Gram=1	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33
Gram=2	0.33	0.33	0.33	0.30	0.23	0.27	0.30	0.33
Gram=3	0.53	0.53	0.47	0.40	0.27	0.23	0.23	0.23
Gram=4	0.57	0.53	0.53	0.43	0.33	0.23	0.20	0.20
Gram=5	0.47	0.50	0.47	0.47	0.33	0.23	0.23	0.27
	Standard error							
Gram=1	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
Gram=2	0.09	0.09	0.09	0.06	0.03	0.03	0.06	0.07
Gram=3	0.13	0.13	0.12	0.10	0.07	0.07	0.07	0.07
Gram=4	0.03	0.07	0.07	0.07	0.13	0.07	0.06	0.10
Gram=5	0.17	0.15	0.18	0.15	0.15	0.03	0.03	0.03

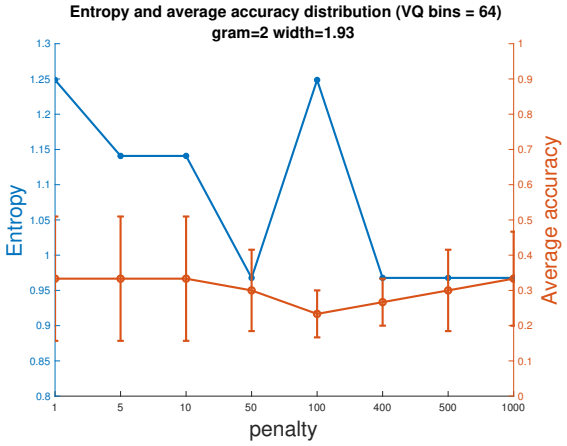
Table 5.3 shows the accuracy and entropy of Cavnar and Trenkle [1994]'s n-gram model with 64 VQ bins. Figure 5.6 compares the accuracies and entropies, the accuracy has error bars ±2 standard error. Considering the highest accuracy and entropy, we find the best performance is the quad-gram (Figure 5.6(d)), whose penalty is 1. And also we conclude that the n-gram still does not work on VLID for 64 VQ bins in most penalty cases since they are guessing randomly between the video languages. The 64 VQ bins case also has the same problem as the 16 VQ bins that, a large histogram binwidth means the language distances can only be binned into a small number of bins which indicates a very spiky histogram - a low entropy. Compare to the 32 VQ bins, the accuracy increases as the 64 VQ bins contains more characters than 32 VQ bins. We find the accuracy decreases with the increasing of penalty for gram 3, 4 and 5, which proves that the bottom-ranked n-grams contain

useless information and confuses the classifier.

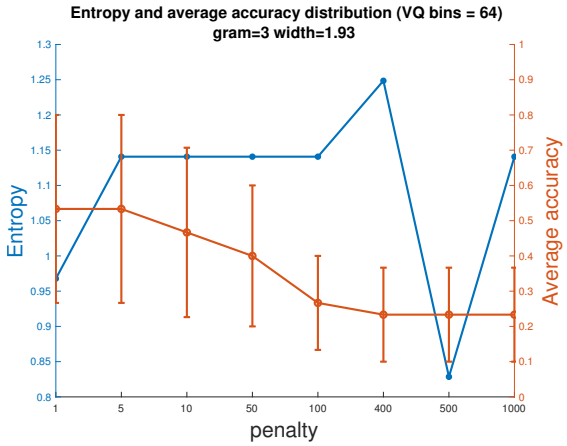
Figure 5.7 visualizes the tri-gram, 10 penalty result in 32 VQ bins. Figure 5.7(a) shows the colour map of languages and Figure 5.7(b) shows the dendrogram which is built based on $d = distance/\sigma$ where d is normalized into $[0, 1]$. The dendrogram is built based on complete-linkage clustering (explained in Section 3.2.2). The video result show in Figure 5.7(a) shows that based on the average distances, English, Arabic and Mandarin are all close to themselves. Since there is no clue in linguistic language tree to present the relationships between English, Mandarin and Arabic, we compare the distances with ALID result with 64 bins. We can find the distances between Arabic and English in Figure 5.7(b) are closer than Arabic and Mandarin while Arabic is more closer to Mandarin in ALID with the same VQ bins (See Figure 4.9(b)). In that case, we think the 64 VQ bins case still performs poorly in VLID.



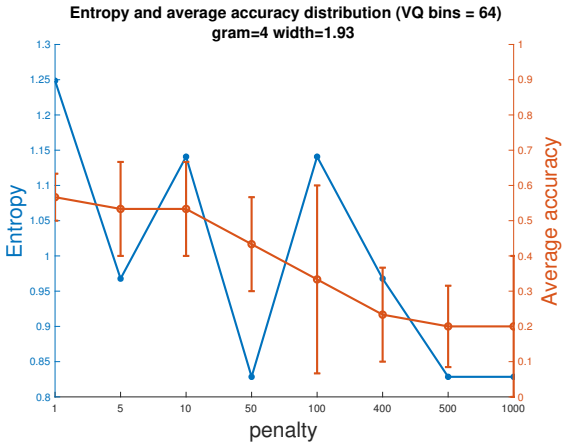
(a) Accuracy and entropy distribution for gram = 1



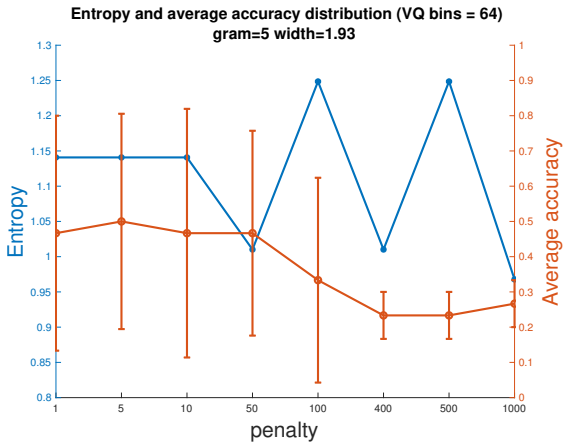
(b) Accuracy and entropy distribution for gram = 2



(c) Accuracy and entropy distribution for gram = 3

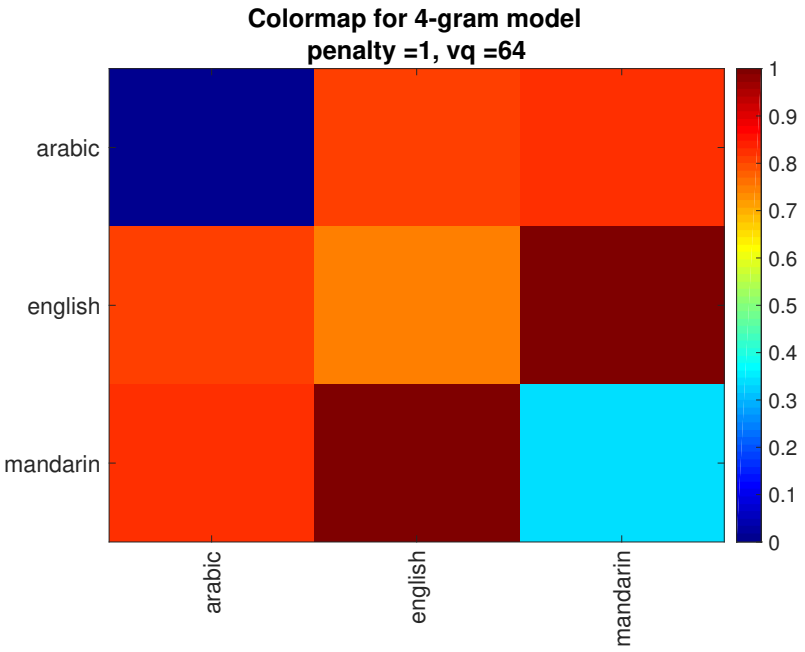


(d) Accuracy and entropy distribution for gram = 4

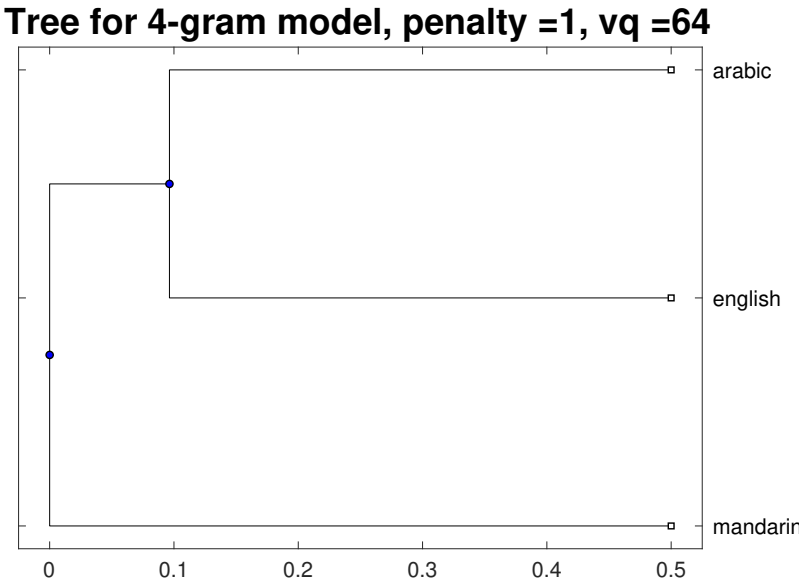


(e) Accuracy and entropy distribution for gram = 5

Figure 5.6: Accuracy and entropy distribution for n-grams. VQ bin size is 64. The x-axis is the penalty value. The left y-axis is the entropy value and the right y-axis is the accuracy value.



(a) Colormap of quad-gram



(b) dendrogram of quad-gram

Figure 5.7: The video language distances results of quad-gram for English, Mandarin and Arabic. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 10 and the VQ bins is 64. Figure 5.7(a) shows the colour map of the language distance variations and Figure 5.7(b) shows the language tree which is built by the distances. The colour variation in Figure 5.7(a) shows the pairwise distances between languages.

5.2.4 Language distance results with 128 bins

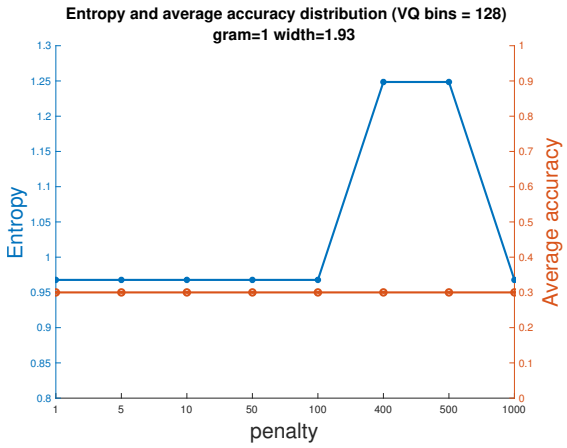
Table 5.4: Entropy(top) and accuracy(bottom) values with histogram binwidth = 1.93, vq bin size = 128.

	Penalty value							
	1	5	10	50	100	400	500	1000
	Entropy value							
Gram=1	0.97	0.97	0.97	0.97	0.97	1.25	1.25	0.97
Gram=2	0.97	1.01	0.83	0.97	1.01	1.01	1.01	1.01
Gram=3	1.14	0.97	0.97	1.25	1.14	0.83	0.83	0.83
Gram=4	0.97	0.97	1.14	0.97	1.01	0.83	0.83	0.83
Gram=5	1.01	1.01	0.97	0.97	0.97	1.14	1.14	1.25
	Accuracy value							
Gram=1	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30
Gram=2	0.60	0.63	0.60	0.47	0.47	0.23	0.27	0.27
Gram=3	0.47	0.47	0.47	0.47	0.50	0.27	0.27	0.37
Gram=4	0.40	0.40	0.40	0.43	0.37	0.33	0.33	0.27
Gram=5	0.50	0.50	0.47	0.40	0.37	0.33	0.33	0.37
	Standard error							
Gram=1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Gram=2	0.06	0.03	0.06	0.07	0.09	0.03	0.03	0.03
Gram=3	0.03	0.03	0.03	0.09	0.10	0.03	0.03	0.09
Gram=4	0.12	0.12	0.12	0.09	0.07	0.03	0.03	0.03
Gram=5	0.15	0.15	0.17	0.10	0.12	0.03	0.03	0.03

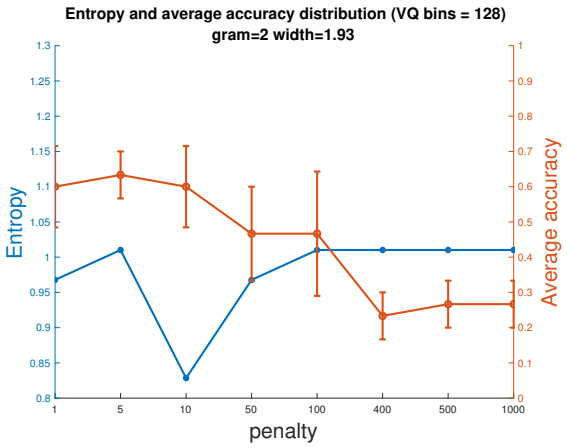
Table 5.4 shows the accuracy and entropy of Cavnar and Trenkle [1994]'s n-gram model with 128 VQ bins. Figure 5.8 compares the accuracies and entropies, the accuracy has error bars ±2 standard error. Considering the highest accuracy and entropy, we find the best performance is the bi-gram (Figure 5.8(b)), whose penalty is 5. And also we conclude that the n-gram still does not work on VLID for 128 VQ bins in most penalty cases since they are guessing randomly between the video languages. The 128 VQ bins case also has the same problem as the 16 VQ bins that, a large histogram binwidth means the language distances can only be binned into a small number of bins which indicates a very spiky histogram - a low entropy. Compare to the 64 VQ bins, the accuracy increases as the 128 VQ bins contains more characters than 64 VQ bins. We find the accuracy decreases with the increasing of penalty for gram 2, 3, 4 and 5, which proves that the bottom-ranked n-grams

contain useless information and confuses the classifier.

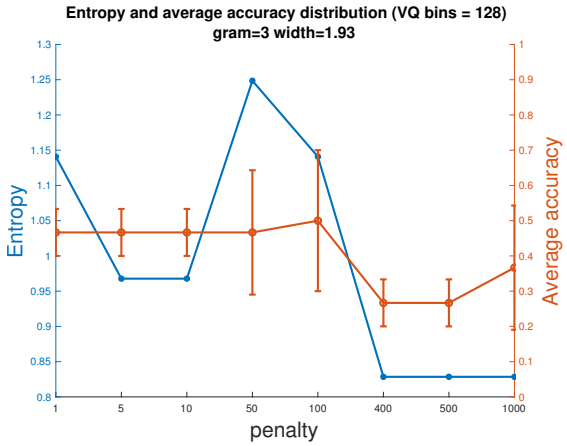
Figure 5.9 visualizes the bi-gram, 5 penalty result in 128 VQ bins. Figure 5.9(a) shows the colour map of languages and Figure 5.9(b) shows the dendrogram which is built based on $d = distance/\sigma$ where d is normalized into $[0, 1]$. The dendrogram is built based on complete-linkage clustering (explained in Section 3.2.2). Figure 5.9(a) shows that based on the average distances, English, Arabic and Mandarin are all close to themselves. Since there is no clue in linguistic language tree to present the relationships between English, Mandarin and Arabic, we compare the distances with ALID result with 128 bins. We can find the distances between Arabic and English in Figure 5.9(b) are closer than Arabic and Mandarin while Arabic is more closer to Mandarin in ALID with the same VQ bins (See Figure 4.11(b)). In that case, we think the 128 VQ bins case performs still performs poorly in VLID.



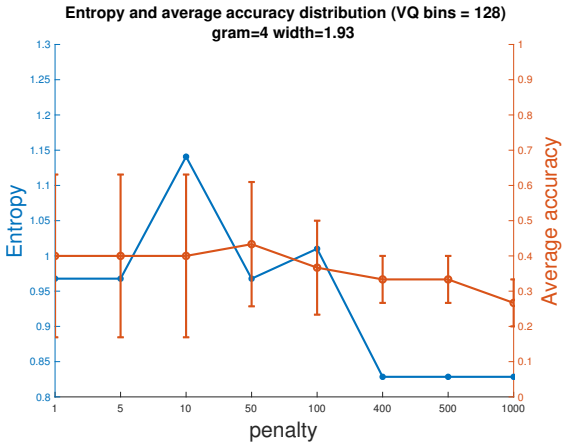
(a) Accuracy and entropy distribution for gram = 1



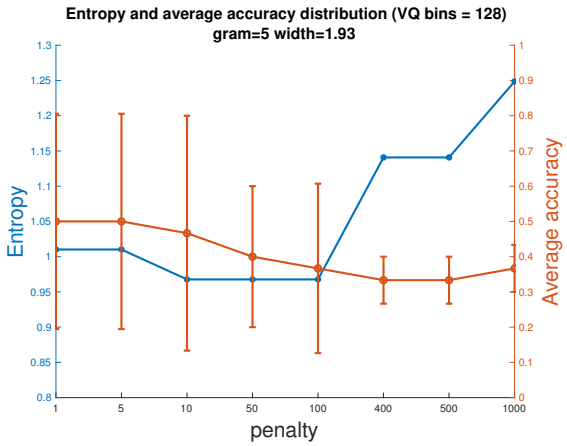
(b) Accuracy and entropy distribution for gram = 2



(c) Accuracy and entropy distribution for gram = 3

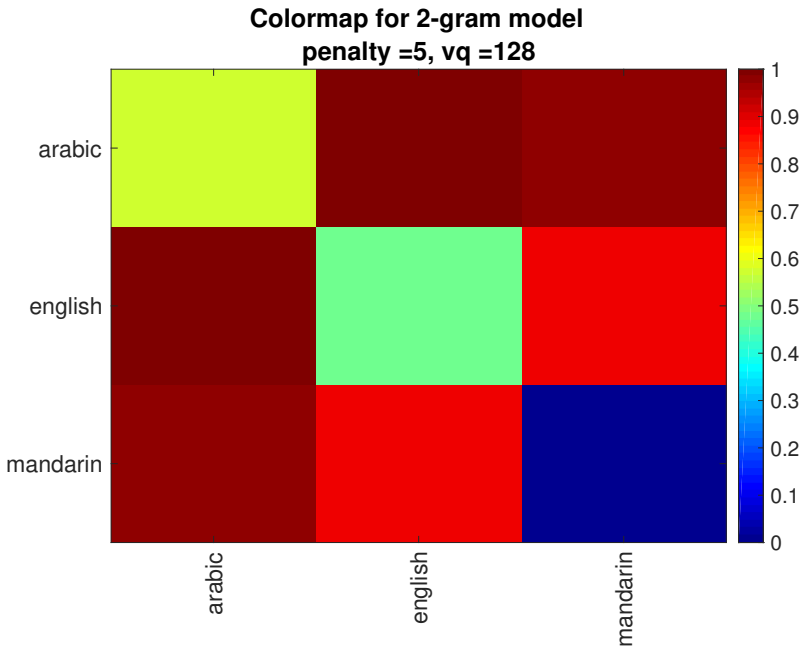


(d) Accuracy and entropy distribution for gram = 4

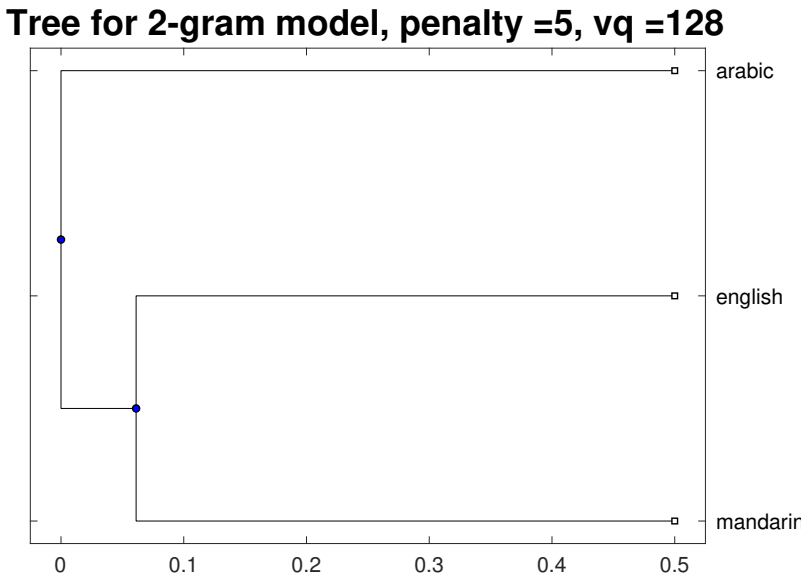


(e) Accuracy and entropy distribution for gram = 5

Figure 5.8: Accuracy and entropy distribution for n-grams. VQ bin size is 128. The x-axis is the penalty value. The left y-axis is the entropy value and the right y-axis is the accuracy value.



(a) Colormap of bi-gram



(b) dendrogram of bi-gram

Figure 5.9: The video language distances results of bi-gram for English, Mandarin and Arabic. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 5 and the VQ bins is 128. Figure 5.9(a) shows the colour map of the language distance variations and Figure 5.9(b) shows the language tree which is built by the distances. The colour variation in Figure 5.9(a) shows the pairwise distances between languages.

5.2.5 Language distance results with 256 bins

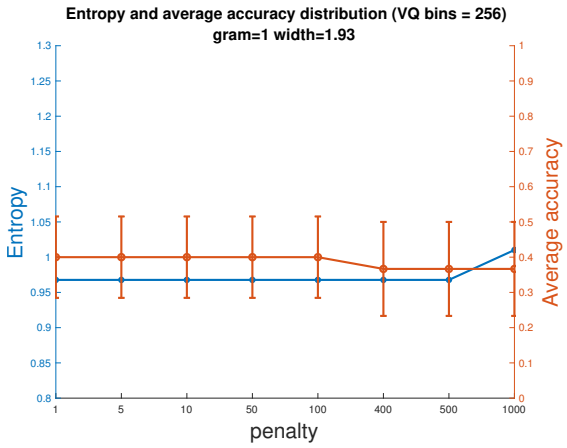
Table 5.5: Entropy(top) and accuracy(bottom) values with histogram binwidth = 1.93, vq bin size = 256.

	Penalty value							
	1	5	10	50	100	400	500	1000
	Entropy value							
Gram=1	0.97	0.97	0.97	0.97	0.97	0.97	0.97	1.01
Gram=2	1.14	0.97	1.01	1.14	1.14	1.25	0.97	0.97
Gram=3	1.14	1.25	1.25	1.14	1.25	1.14	1.14	1.25
Gram=4	0.83	1.14	1.14	0.83	1.14	1.25	1.25	0.83
Gram=5	1.14	1.14	0.97	1.25	1.25	0.83	0.83	0.83
	Accuracy value							
Gram=1	0.40	0.40	0.40	0.40	0.40	0.37	0.37	0.37
Gram=2	0.47	0.47	0.43	0.40	0.37	0.37	0.40	0.33
Gram=3	0.43	0.43	0.43	0.40	0.43	0.27	0.23	0.27
Gram=4	0.40	0.40	0.40	0.37	0.30	0.27	0.27	0.30
Gram=5	0.30	0.30	0.30	0.27	0.23	0.27	0.30	0.30
	Standard error							
Gram=1	0.06	0.06	0.06	0.06	0.06	0.07	0.07	0.07
Gram=2	0.07	0.07	0.09	0.00	0.12	0.17	0.15	0.09
Gram=3	0.03	0.03	0.03	0.00	0.09	0.07	0.03	0.03
Gram=4	0.06	0.06	0.06	0.09	0.12	0.12	0.12	0.10
Gram=5	0.06	0.06	0.06	0.07	0.03	0.09	0.12	0.15

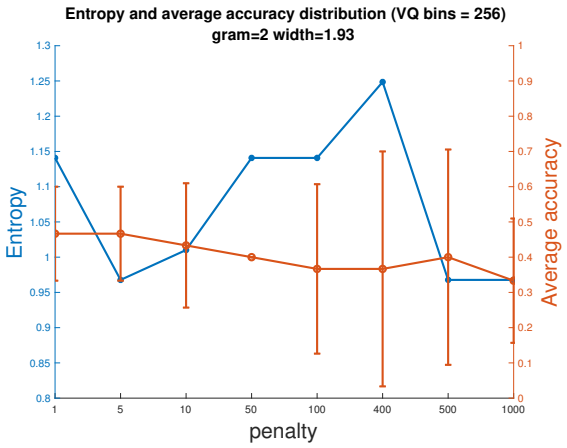
Table 5.5 shows the accuracy and entropy of Cavnar and Trenkle [1994]'s n-gram model with 256 VQ bins. Figure 5.10 compares the accuracies and entropies, the accuracy has error bars ±2 standard error. Considering the highest accuracy and entropy, we find the best performance is the bi-gram (Figure 5.10(b)), whose penalty is 1. As we previously explained, the 256 VQ bins lose more data information, thus it is not surprising that 256 VQ bins case produces a lower accuracy than 128 VQ bins. And also we conclude that the n-gram still does not work on VLID for 256 VQ bins in most penalty cases since they are guessing randomly between the video languages.

Figure 5.11 visualizes the bi-gram, 5 penalty result in 256 VQ bins. Figure 5.11(a) shows the colour map of languages and Figure 5.11(b) shows the dendrogram which is built based on $d = distance/\sigma$ where d is normalized into $[0, 1]$. The dendrogram

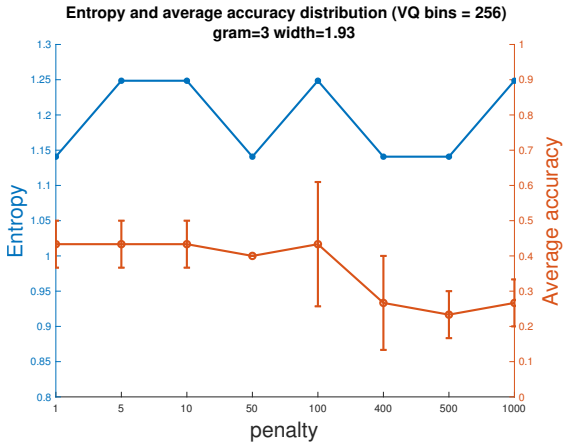
is built based on complete-linkage clustering (explained in Section 3.2.2). Figure 5.11(a) shows that English is closer to Arabic than itself. Since there is no clue in linguistic language tree to present the relationships between English, Mandarin and Arabic, we compare the distances with ALID result with 256 bins. We can find the distances between Arabic and English in Figure 5.11(b) are closer than Arabic and Mandarin while Arabic is more closer to Mandarin in ALID with the same VQ bins (See Figure 4.13(b)). In that case, we think the 256 VQ bins case still performs poorly in VLID.



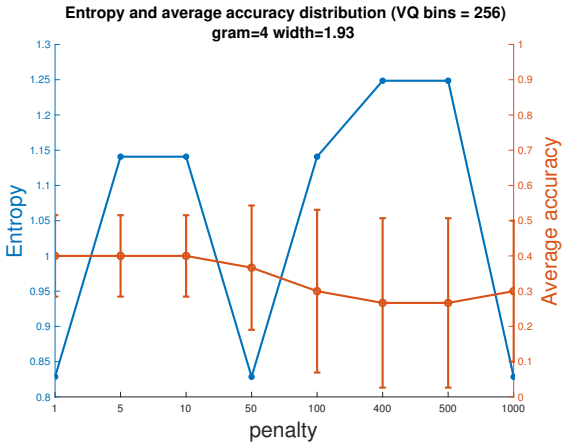
(a) Accuracy and entropy distribution for gram = 1



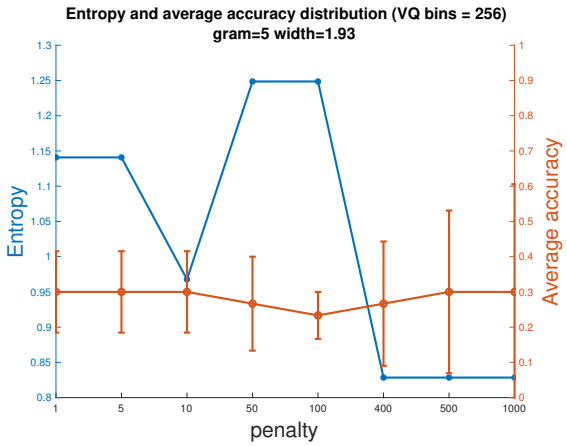
(b) Accuracy and entropy distribution for gram = 2



(c) Accuracy and entropy distribution for gram = 3

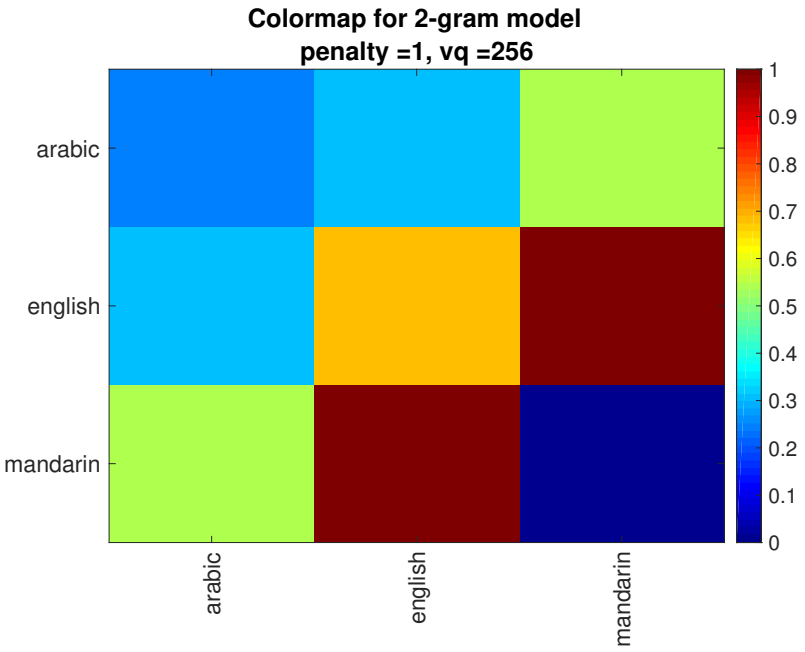


(d) Accuracy and entropy distribution for gram = 4

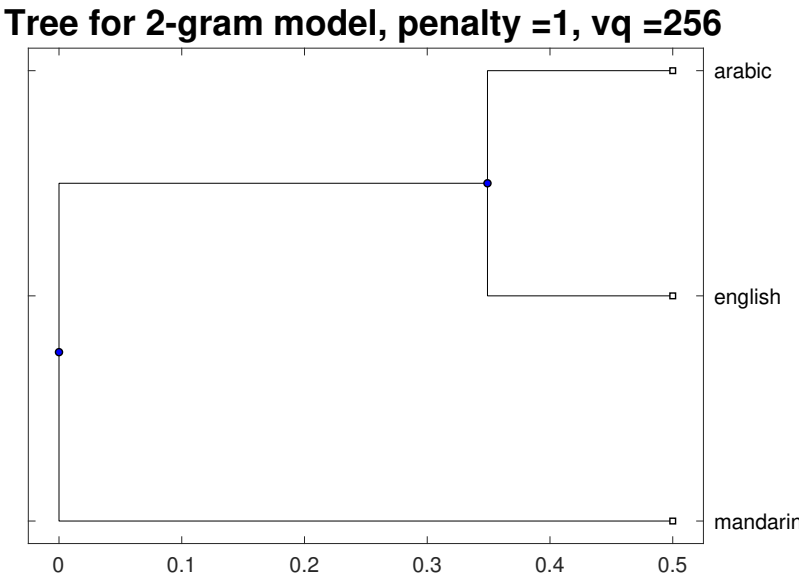


(e) Accuracy and entropy distribution for gram = 5

Figure 5.10: Accuracy and entropy distribution for n-grams. VQ bin size is 256. The x-axis is the penalty value. The left y-axis is the entropy value and the right y-axis is the accuracy value.



(a) Colormap of bi-gram



(b) dendrogram of bi-gram

Figure 5.11: The video language distances results of bi-gram for English, Mandarin and Arabic. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 1 and the VQ bins is 256. Figure 5.11(a) shows the colour map of the language distance variations and Figure 5.11(b) shows the language tree which is built by the distances. The colour variation in Figure 5.11(a) shows the pairwise distances between languages.

5.2.6 Conclusion

Based on Cavnar and Trenkle [1994]'s n-gram VLID results, we can see that the accuracy in VLID is not as good as ALID. The entropy of histogram is also not able to fully describe how distinctive that languages distances are. This is probably because of the lack of data in video dataset (which only have 3 languages with a small number of speakers). Also, the gesture made by the speakers impact on the recognition results. For example, two Mandarin speakers are not moving their mouths obviously and their VQ strings are, hence, full of repeat symbols. However, we still can conclude that for VLID, the best performance is in 64 and 128 VQ bins. It is possible that the relationships between these three languages will be more clear if there is enough data.

5.3 Compression distances by zipping

5.3.1 Methods

This section applies zipping methods to the VLID system. We use the three compressors discussed in previous chapters: zip in 3.3.1.1, bzip in 3.3.2.1 and ppm in 3.3.3.1. We use the vector quantisation by the same procedure as in 4.4 VLID features - the AAMs. VQ then converts AAMs into Unicode characters. In zipping, we also wonder whether VQ binsize impacts on the results. In this case, we examine the compression results on 16, 32, 64, 128 and 256 bins, which is the same as n-gram method.

For these results, 0 following the name of the compressor denotes the interleaving status. For example, zip0, means non-interleaved string with zip compressor and ppm1 means an interleaved string with a ppm compressor.

5.3.2 Language distance results with 16 bins

This section describes the video language distances by using colour maps, phylogenetic trees and histogram distributions. The number of VQ bins is 16. The description of phylogenetic tree is in Section 3.2.2 and the description of histogram distribution is in Section 3.2.1.2. Figure 5.12 to 5.14 show the colour maps of the languages distances. Figure 5.15 to 5.17 show the dendrograms of language distances.

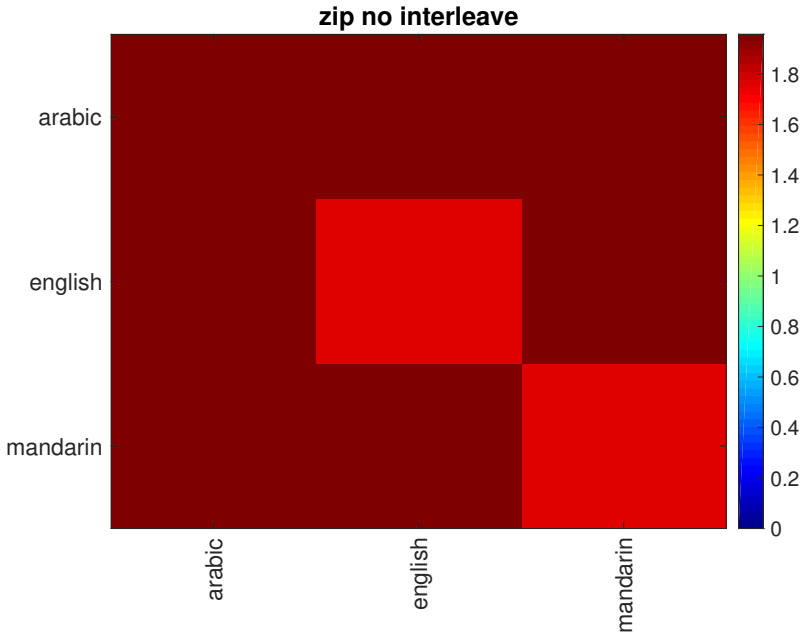
Table 5.6: Entropy(top) and accuracy(bottom) values with histogram binwidth = 1.93, vq binsize = 16.

	Zppm0	Zppm1	Zzip0	Zzip1	Zbzip0	Zbzip1
Entropy	1.00	0.65	0.92	1.00	1.00	1.00
Accuracy	1.00	1.00	1.00	1.00	1.00	1.00

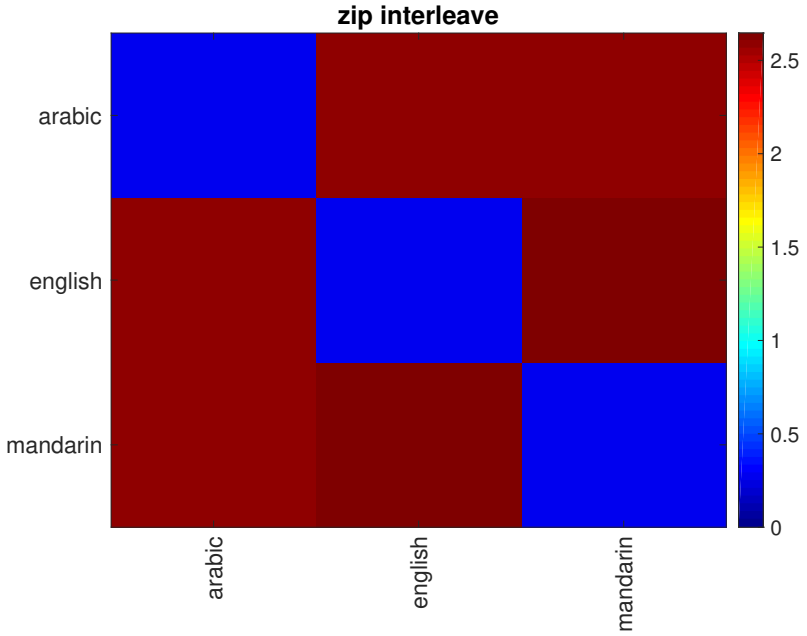
Table 5.6 concludes the entropy values of the histogram distributions for ppm, zip and bzip with interleaved and non-interleaved data. The results show the recognition accuracies of all compressions are 100% and the highest entropy is 1.00. According to Equation 3.10, the distances of the languages are calculated by analysing the compressed length of the strings. For measuring the distance of language itself, the zipping method compresses one string with itself. So the ppm, bzip and zip do not need to predict the characters which have never been seen before. Thus, the compression entropy of language itself is always the smallest and the recognition accuracy is always 100%. For reference, a histogram with two equiprobable bins would have an entropy of 1 bit whereas a 16 VQ binsize of the histogram with equiprobable bins would have an entropy of 4 bits. Thus 1 bit indicates a very non-smooth histogram (an all-or-nothing distance).

Figure 5.12 to Figure 5.17 show the colour maps and dendrograms of the pairwise language distances for each compression with interleaved and non-interleaved data. For zip, bzip and ppm, although we can see the interleaving result shows good performances of language identification, both the interleaving and the non-interleaving result can hardly show the distances relationships between the languages and the variation is much lower than Cavnar and Trenkle [1994]'s n-gram model.

The ppm (Figure 5.13) and bzip (Figure 5.14) show the same problem as zip results. Like the VLID n-gram results, we compare the VLID zipping trees with ALID ones under the same VQ binsize. For zip interleaving and non-interleaving result, the ALID (Figure 4.17) shows Arabic is close to Mandarin while the VLID result shows English is closer to Mandarin in the non-interleaving result and closer to Arabic in the interleaving result. For bzip non-interleaving result, the ALID (Figure 4.15(a)) shows Arabic is close to Mandarin while the VLID result shows English is closer to Arabic. The ALID interleaving result (Figure 4.15(b)) shows that Mandarin is close to English while VLID result shows the same as the non-interleaving result that English is closer to Arabic. For ppm, the ALID results (Figure 4.19), both interleaving and non-interleaving results tell Mandarin and Arabic are close while the VLID results show the Arabic and English are similar. In that case, we think the 16 VQ bins case performs poorly in VLID.

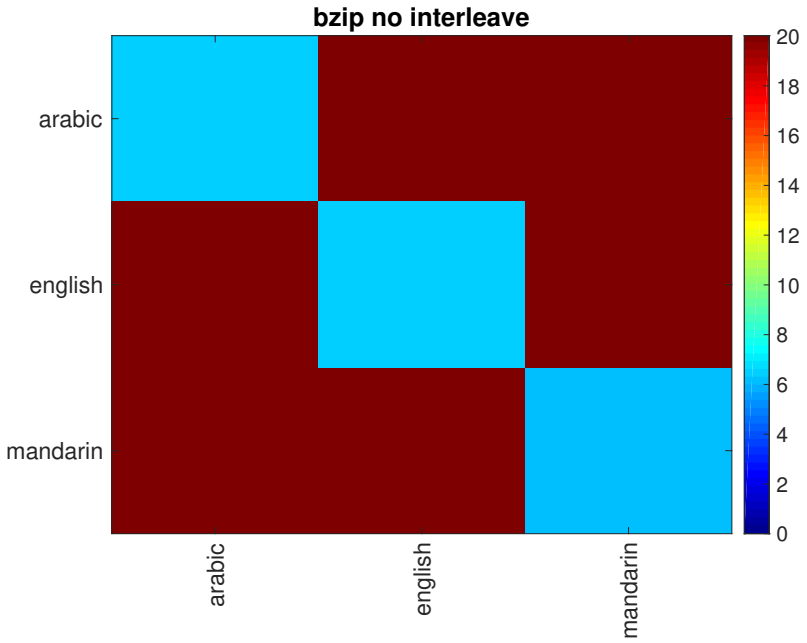


(a) without interleave

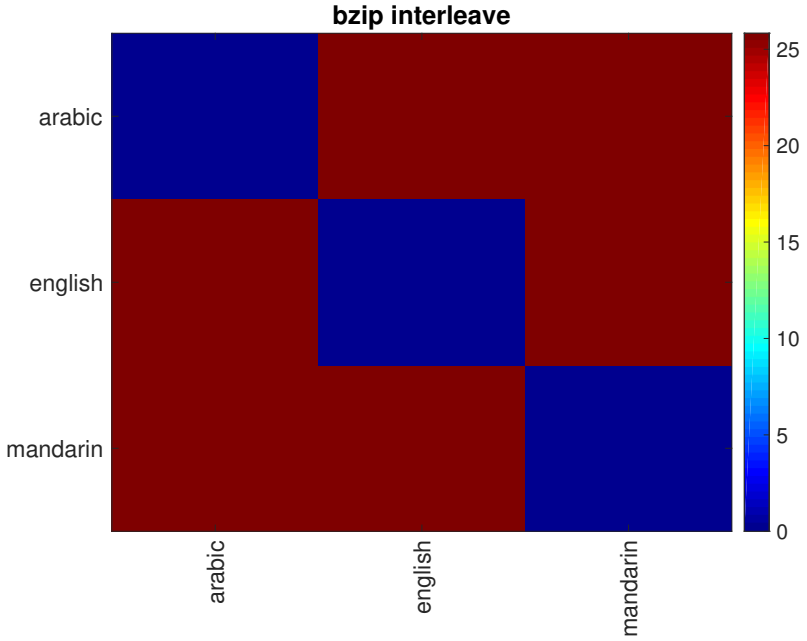


(b) with interleave

Figure 5.12: The 21 UNDHR video languages distances are computed by zip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 5.12(a) shows the non-interleaved result and Figure 5.12(b) shows the interleaved result.

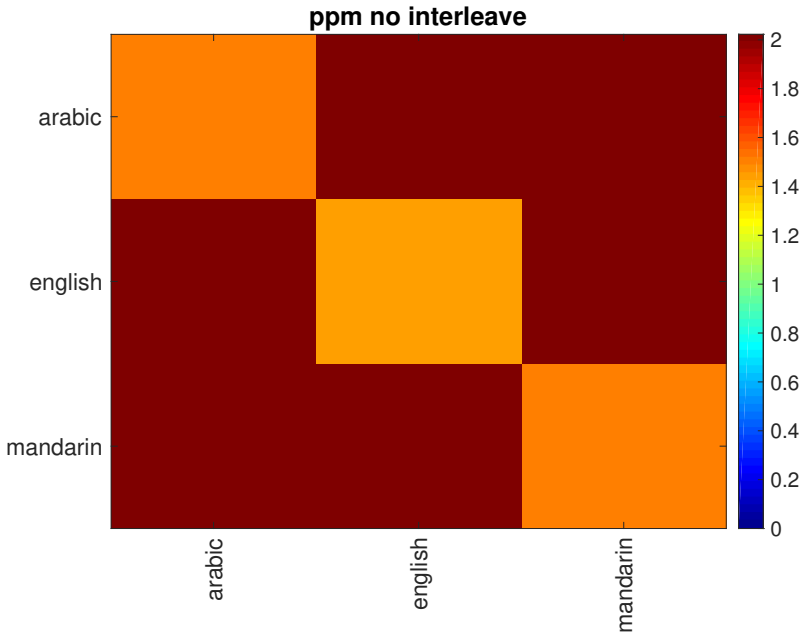


(a) without interleave

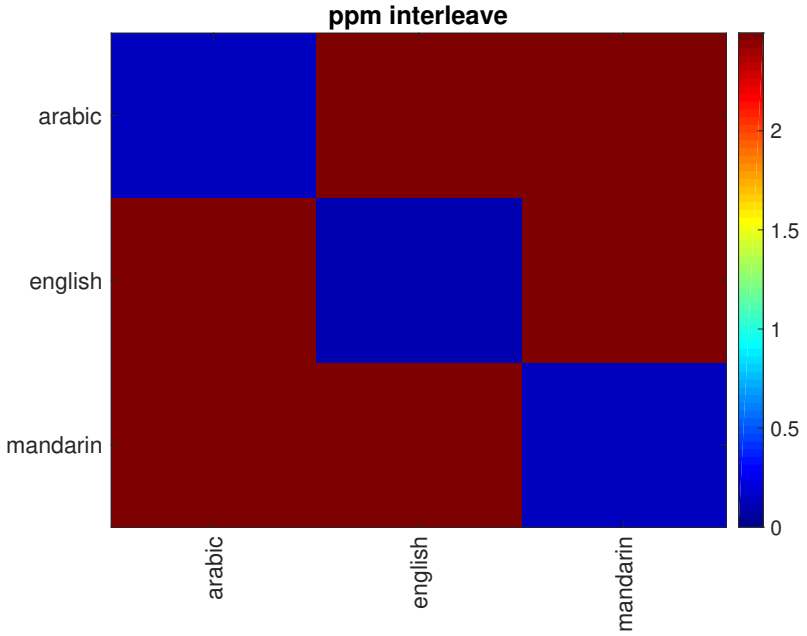


(b) with interleave

Figure 5.13: The 21 UNDHR video languages distances are computed by bzip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 5.12(a) shows the non-interleaved result and Figure 5.13(b) shows the interleaved result.

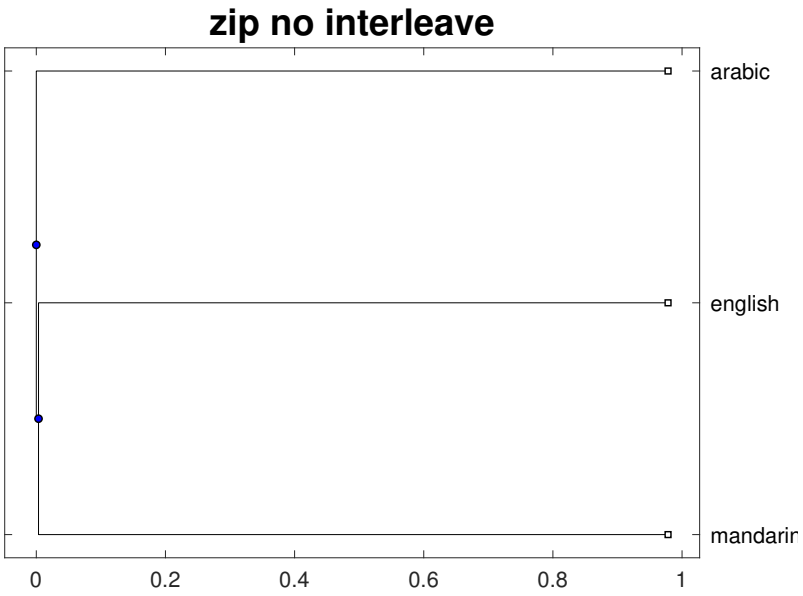


(a) without interleave

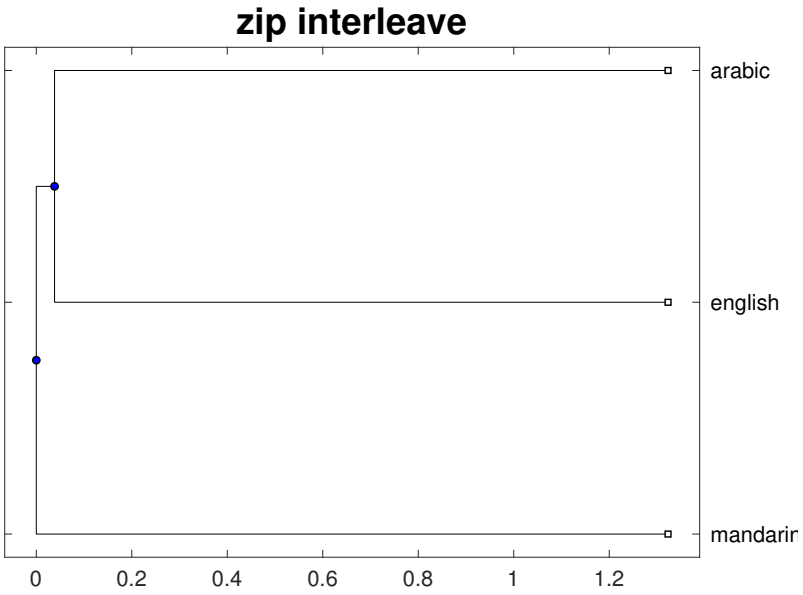


(b) with interleave

Figure 5.14: The 21 UNDHR video languages distances are computed by ppm and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 5.14(a) shows the non-interleaved result and Figure 5.14(b) shows the interleaved result.

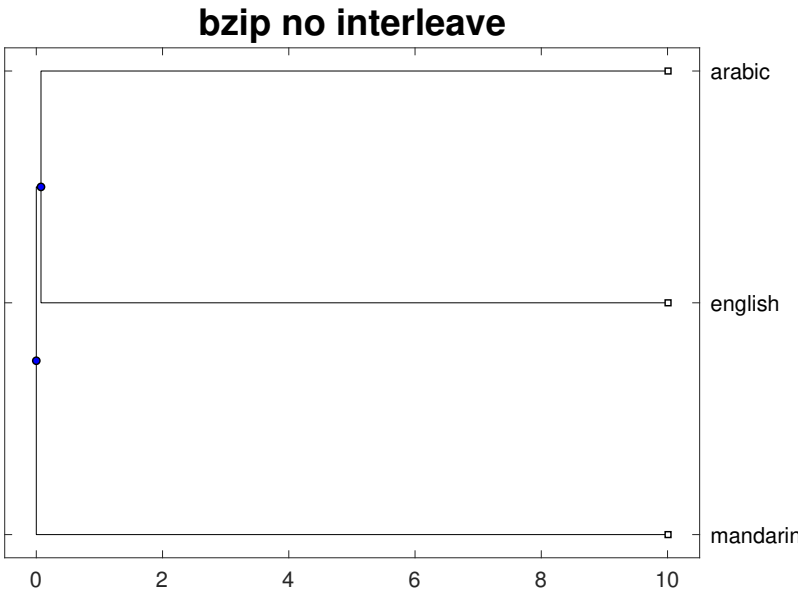


(a) without interleave

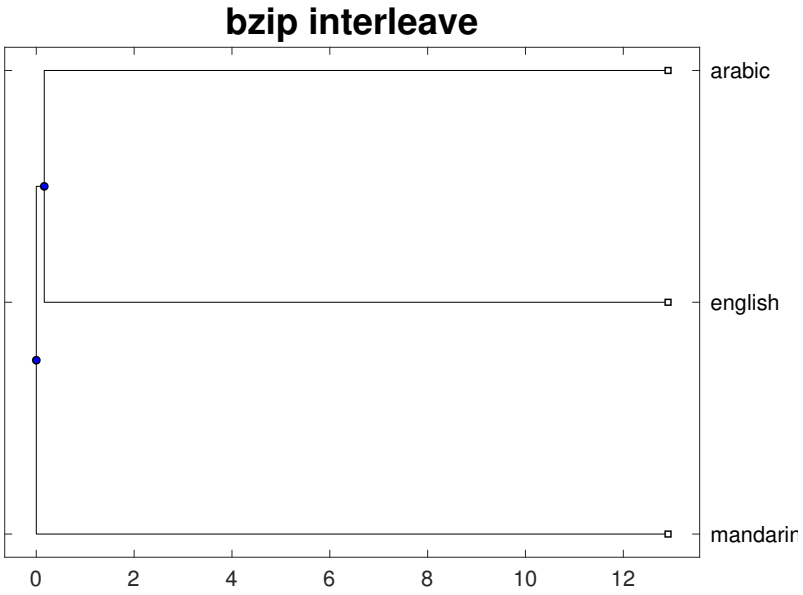


(b) with interleave

Figure 5.15: The 21 UNDHR video languages distances are computed by zip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 5.15(a) shows the non-interleaved result and Figure 5.15(b) shows the interleaved result.

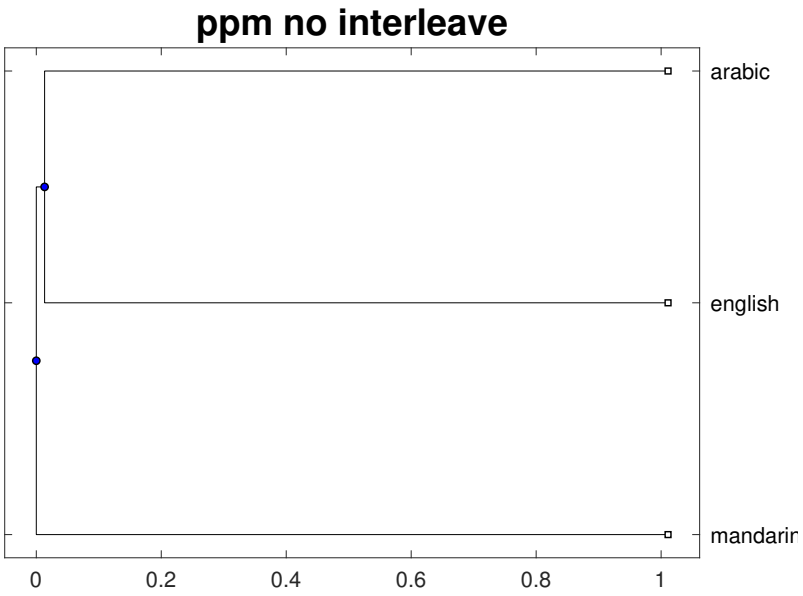


(a) without interleave

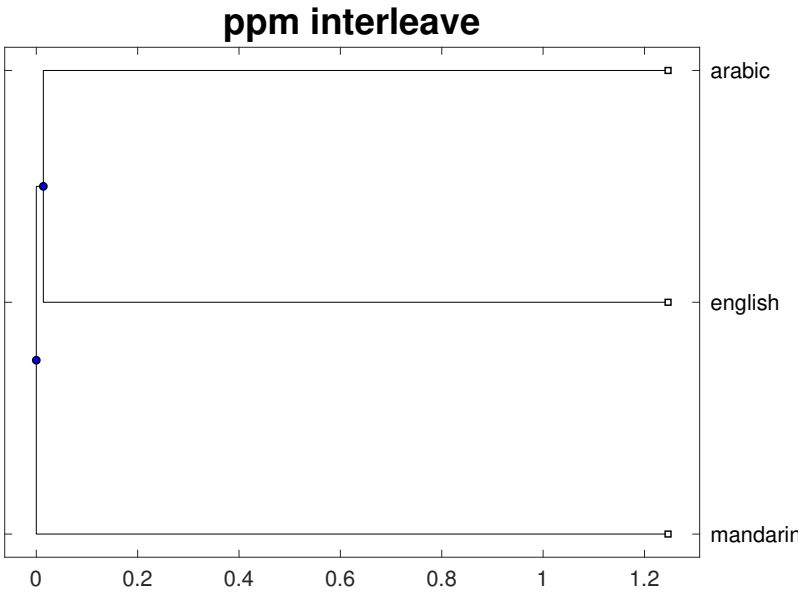


(b) with interleave

Figure 5.16: The 21 UNDHR video languages distances are computed by bzip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 5.16(a) shows the non-interleaved result and Figure 5.16(b) shows the interleaved result.



(a) without interleave



(b) with interleave

Figure 5.17: The 21 UNDHR video languages distances are computed by ppm and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 16. Figure 5.17(a) shows the non-interleaved result and Figure 5.17(b) shows the interleaved result.

5.3.3 Language distance results with 32 bins

This section describes the video language distances by using colour maps, phylogenetic trees and histogram distributions. The number of VQ bins is 32. The description of phylogenetic tree is in Section 3.2.2 and the description of histogram distribution is in Section 3.2.1.2. Figure 5.18 to 5.20 show the colour maps of the languages distances. Figure 5.21 to 5.23 show the dendrograms of language distances.

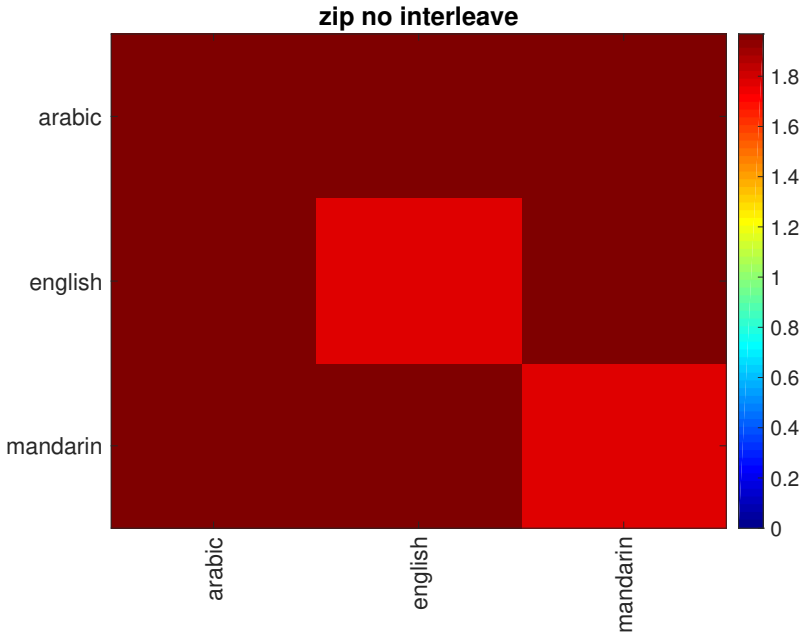
Table 5.7: Entropy(top) and accuracy(bottom) values with histogram binwidth = 1.93, vq binsize = 32.

	Zppm0	Zppm1	Zzip0	Zzip1	Zbzip0	Zbzip1
Entropy	1.00	0.65	1.46	1.00	1.00	1.00
Accuracy	1.00	1.00	1.00	1.00	1.00	1.00

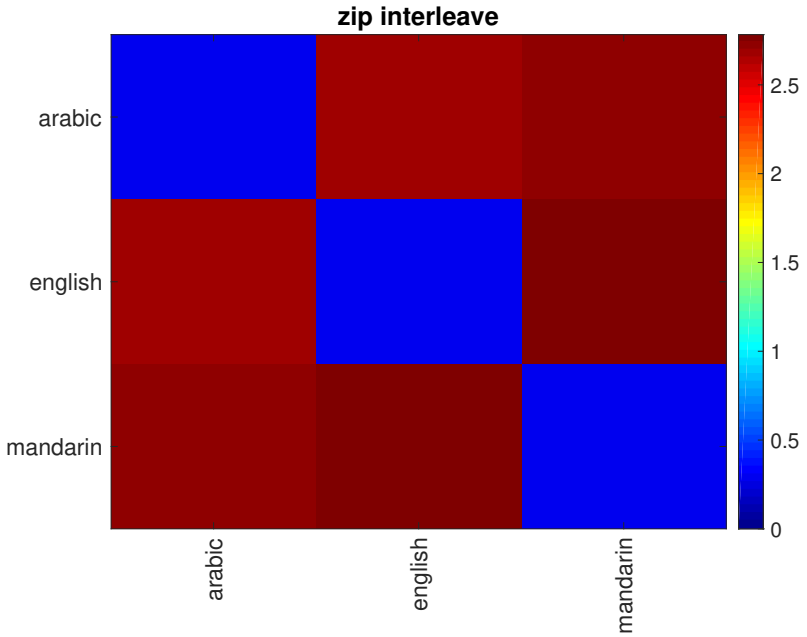
Table 5.7 concludes the entropy values of the histogram distributions for ppm, zip and bzip with interleaved and non-interleaved data. The results show the recognition accuracies of all compressions are 100% and the highest entropy is 1.46. There are still some entropy values are calculated as 1. For the same reason as 16 VQ bins, a histogram with two equiprobable bins would have an entropy of 1 bit. Thus 1 bit indicates a very non-smooth histogram (an all-or-nothing distance).

Figure 5.18 to Figure 5.23 show the colour maps and dendrograms of the pairwise language distances for each compression with interleaved and non-interleaved data. For zip, bzip and ppm, although we can see the interleaving result also shows good performances of language identification, both the interleaving and the non-interleaving result still can hardly tell the distances relationships between the languages and the variation is also lower than Cavnar and Trenkle [1994]'s n-gram model. The ppm (Figure 5.19) and bzip (Figure 5.20) show the same problem as the zip results. Like the VLID n-gram results, we compare the VLID zipping trees with ALID ones under the same VQ binsize. For zip, bzip and ppm interleaving and non-interleaving result, the ALID results (Figure 4.23, 4.24, 4.25) show that Arabic is close to Mandarin while the VLID result in zip shows English is more closer to Mandarin in the zip interleaving result and is more closer to Arabic in the other

results. In that case, we think the 32 VQ bins case also performs poorly in VLID.

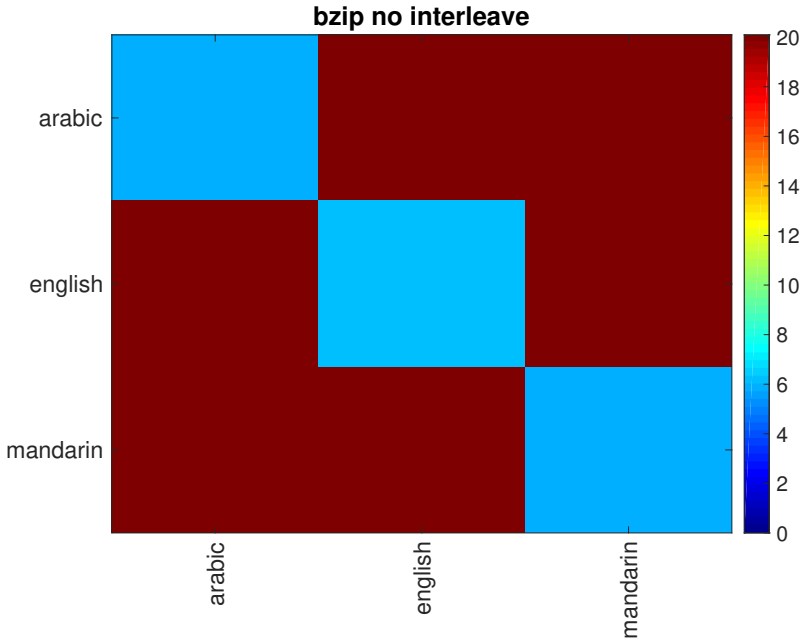


(a) without interleave

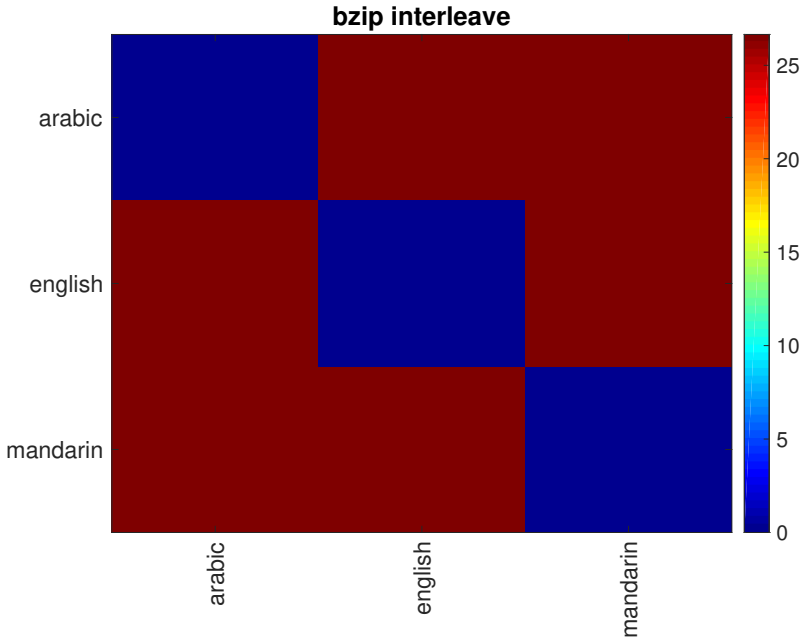


(b) with interleave

Figure 5.18: The 21 UNDHR video languages distances are computed by zip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 5.18(a) shows the non-interleaved result and Figure 5.18(b) shows the interleaved result.

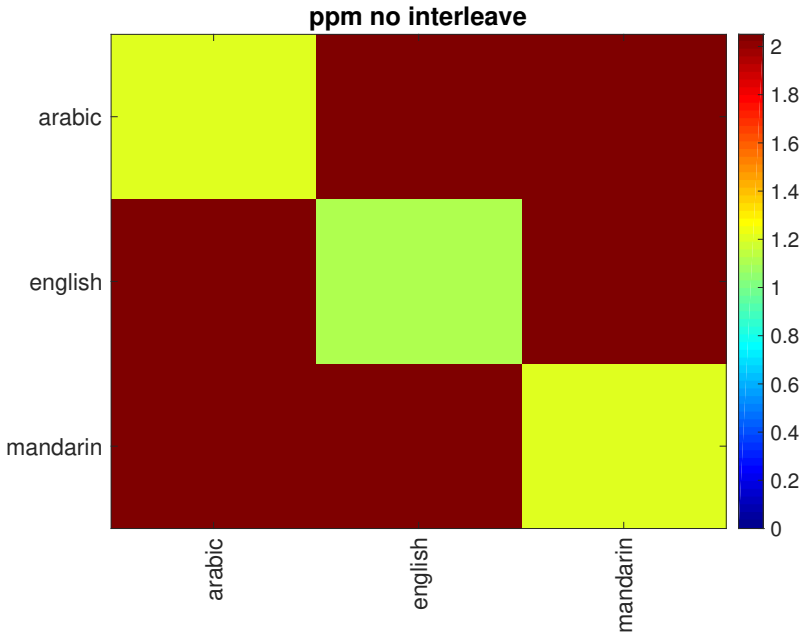


(a) without interleave

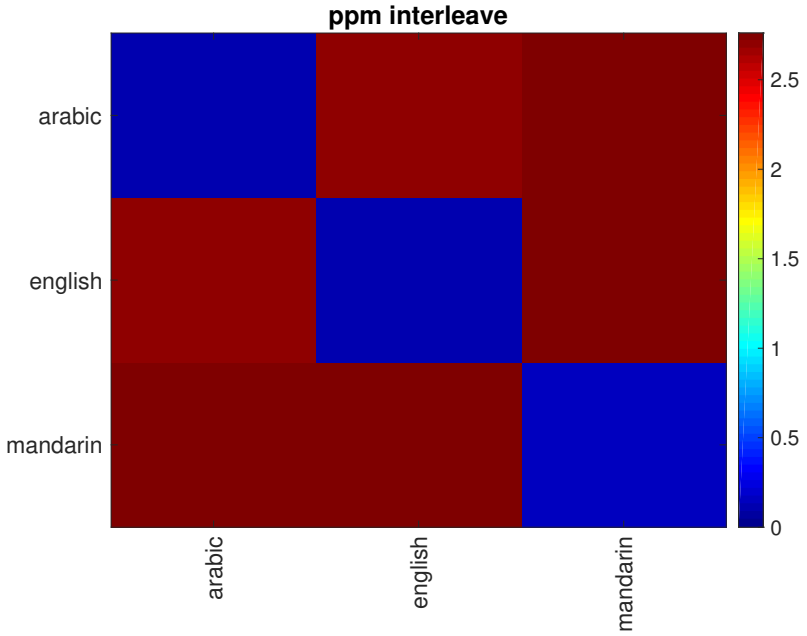


(b) with interleave

Figure 5.19: The 21 UNDHR video languages distances are computed by bzip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 5.18(a) shows the non-interleaved result and Figure 5.19(b) shows the interleaved result.

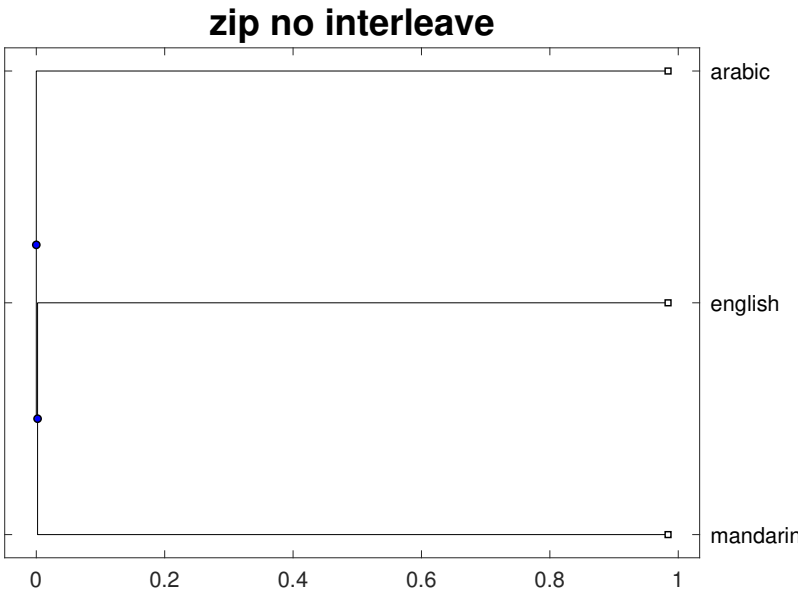


(a) without interleave

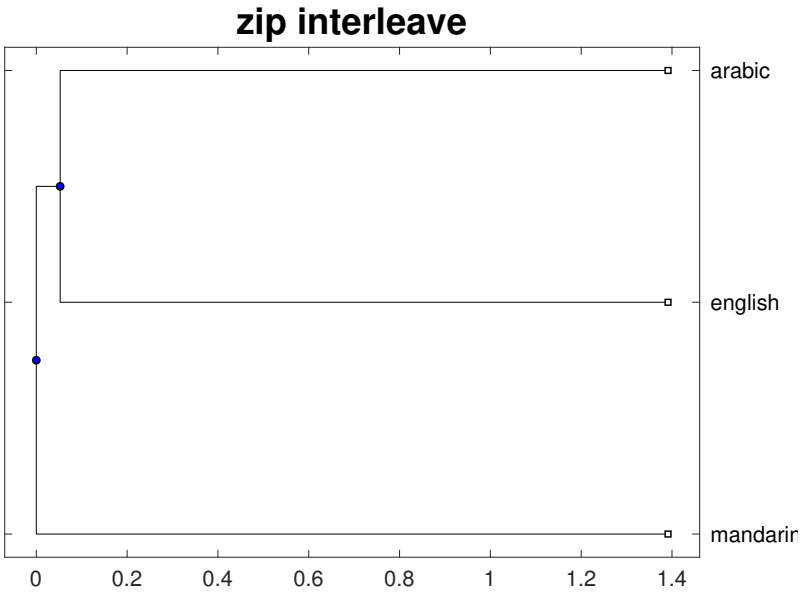


(b) with interleave

Figure 5.20: The 21 UNDHR video languages distances are computed by ppm and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 5.20(a) shows the non-interleaved result and Figure 5.20(b) shows the interleaved result.

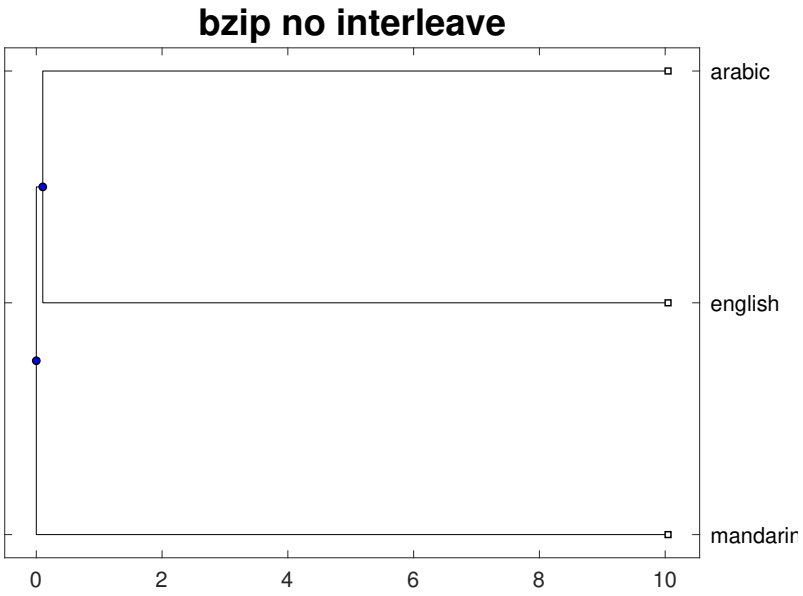


(a) without interleave

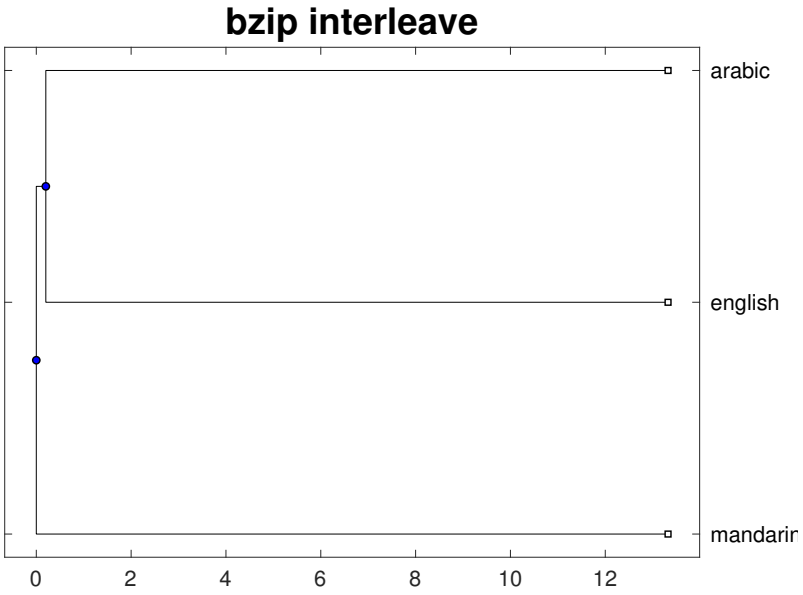


(b) with interleave

Figure 5.21: The 21 UNDHR video languages distances are computed by zip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 5.21(a) shows the non-interleaved result and Figure 5.21(b) shows the interleaved result.

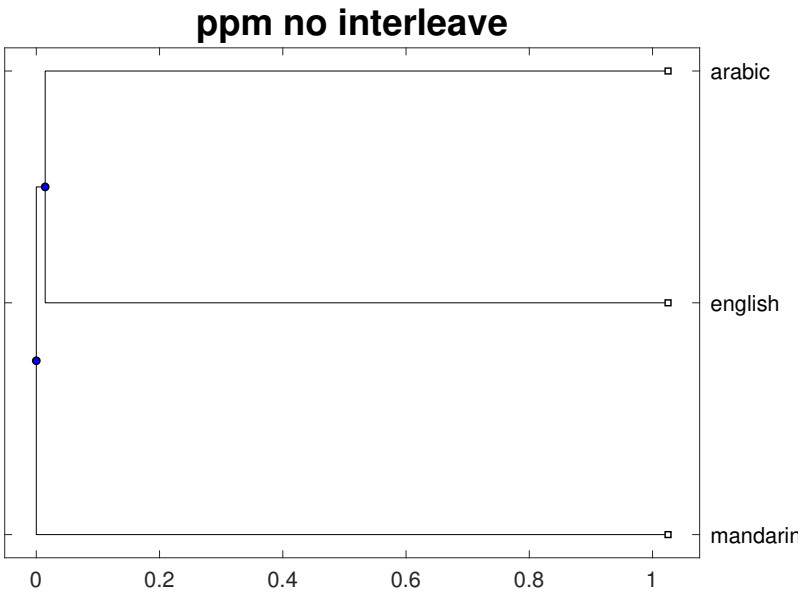


(a) without interleave

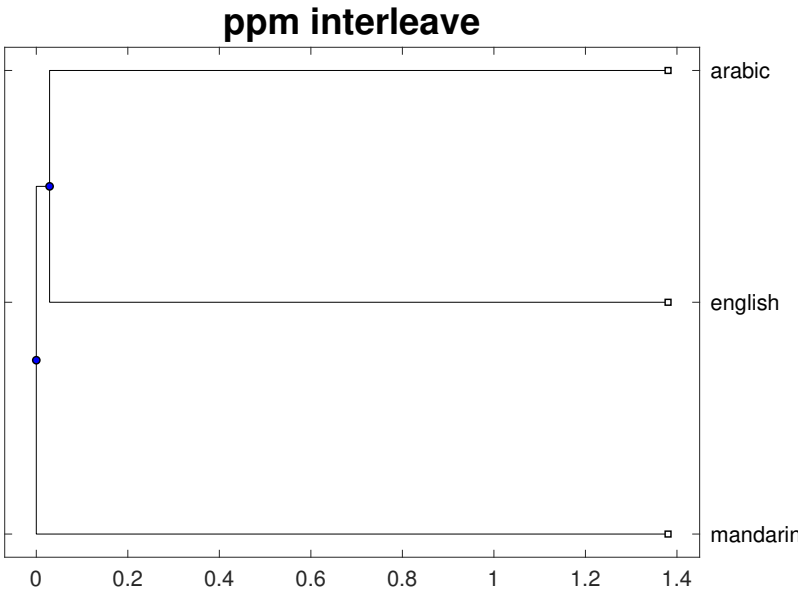


(b) with interleave

Figure 5.22: The 21 UNDHR video languages distances are computed by bzip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 5.22(a) shows the non-interleaved result and Figure 5.22(b) shows the interleaved result.



(a) without interleave



(b) with interleave

Figure 5.23: The 21 UNDHR video languages distances are computed by ppm and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 32. Figure 5.23(a) shows the non-interleaved result and Figure 5.23(b) shows the interleaved result.

5.3.4 Language distance results with 64 bins

This section describes the video language distances by using colour maps, phylogenetic trees and histogram distributions. The number of VQ bins is 64. The description of phylogenetic tree is in Section 3.2.2 and the description of histogram distribution is in Section 3.2.1.2. Figure 5.24 to 5.26 show the colour maps of the languages distances. Figure 5.27 to 5.29 show the dendrograms of language distances.

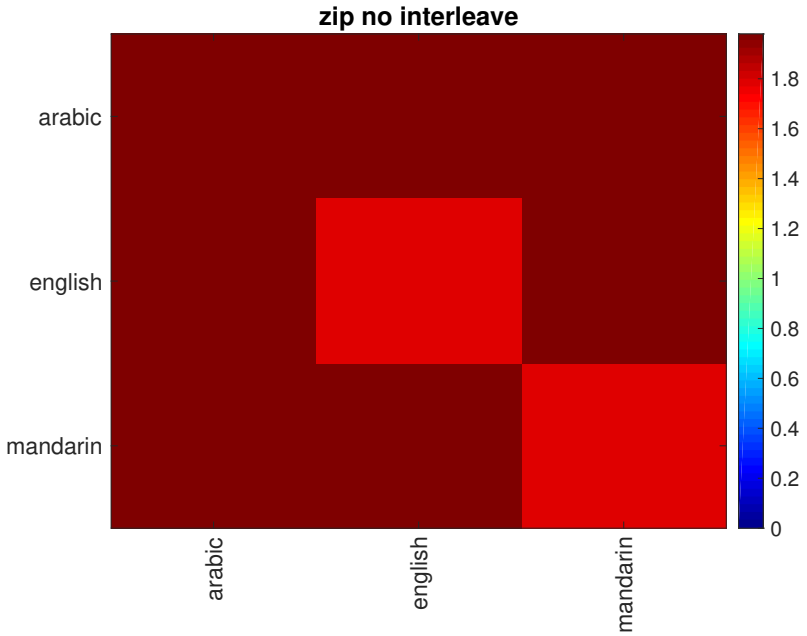
Table 5.8: Entropy(top) and accuracy(bottom) values with histogram binwidth = 1.93, vq binsize = 64.

	Zppm0	Zppm1	Zzip0	Zzip1	Zbzip0	Zbzip1
Entropy	1.00	0.65	0.92	1.00	1.00	0.00
Accuracy	1.00	1.00	1.00	1.00	1.00	1.00

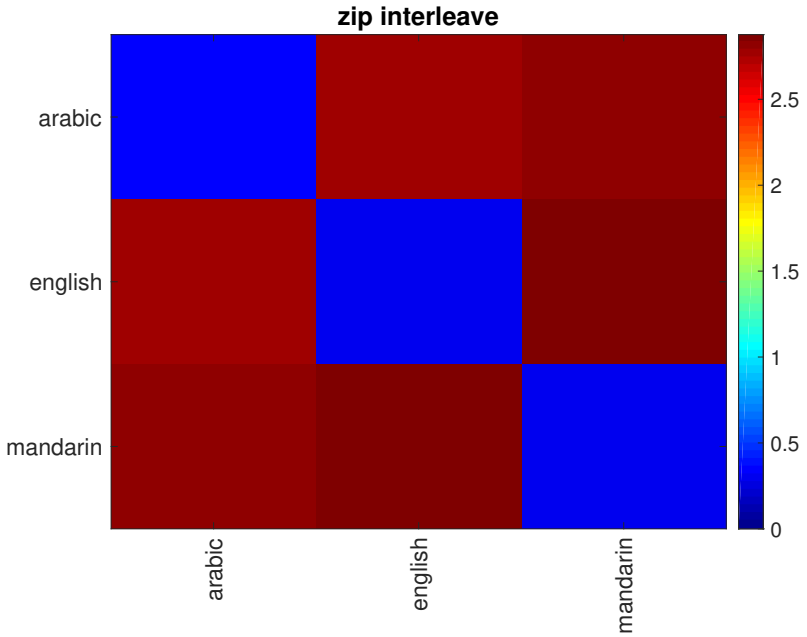
Table 5.8 concludes the entropy values of the histogram distributions for ppm, zip and bzip with interleaved and non-interleaved data. The results show the recognition accuracies of all compressions are 100% and the highest entropy is 1. For the same reason as 16 VQ bins, a histogram with two equiprobable bins would have an entropy of 1 bit. Thus 1 bit in this case still indicates a very non-smooth histogram (an all-or-nothing distance). And if the histogram with one bin would have an entropy of 0 bit, which means the language distances variation is not distinctive.

Figure 5.24 to Figure 5.29 show the colour maps and dendrograms of the pair-wise language distances for each compression with interleaved and non-interleaved data. For zip, bzip and ppm, like 32 VQ bins, we still can see the interleaving result shows good performances of language identification while both the interleaving and the non-interleaving result can hardly show the distances relationships between the languages and the variation is much lower than Cavnar and Trenkle [1994]'s n-gram model. The ppm (Figure 5.26) and bzip (Figure 5.25) show the same problem as the zip results. Like the VLID n-gram results, we compare the VLID zipping trees with ALID ones under the same VQ binsize. For zip, bzip and ppm interleaving and non-interleaving result, the ALID results (Figure 4.29, 4.30, 4.31) show that Arabic is close to Mandarin while the VLID result in zip shows English is more closer to

Arabic. In that case, we think the 64 VQ bins case also performs poorly in VLID.

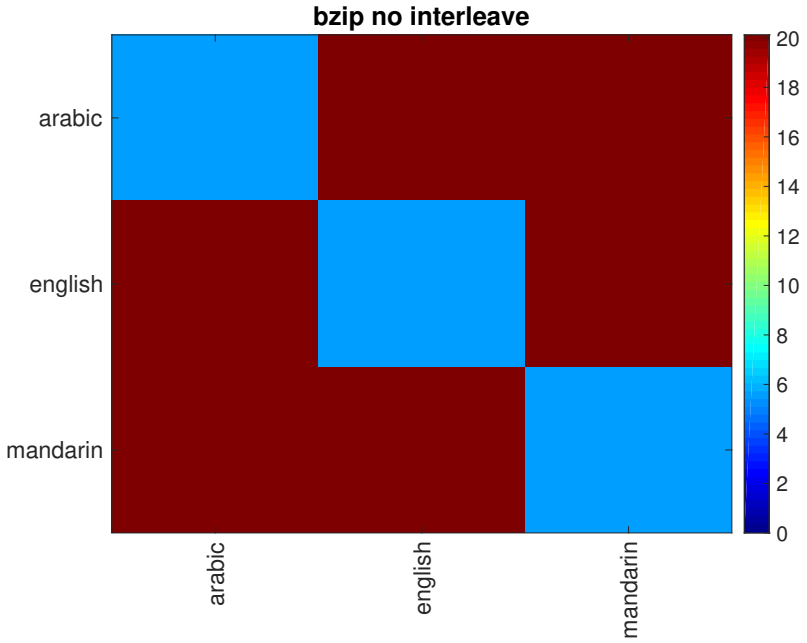


(a) without interleave

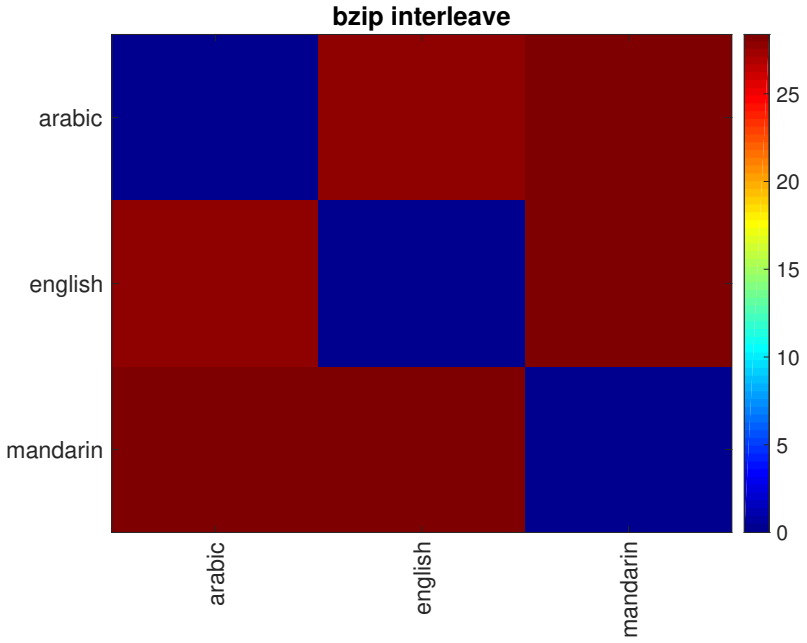


(b) with interleave

Figure 5.24: The 21 UNDHR video languages distances are computed by zip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 5.24(a) shows the non-interleaved result and Figure 5.24(b) shows the interleaved result.

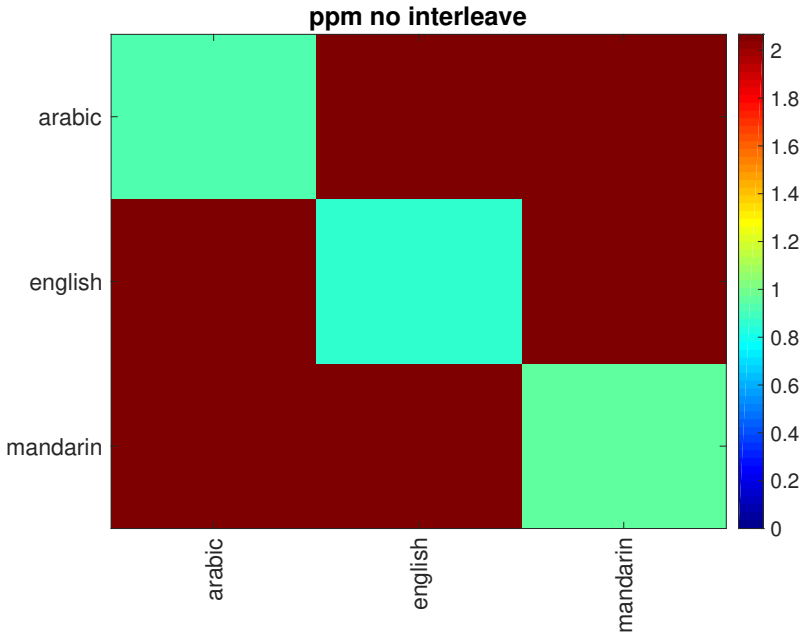


(a) without interleave

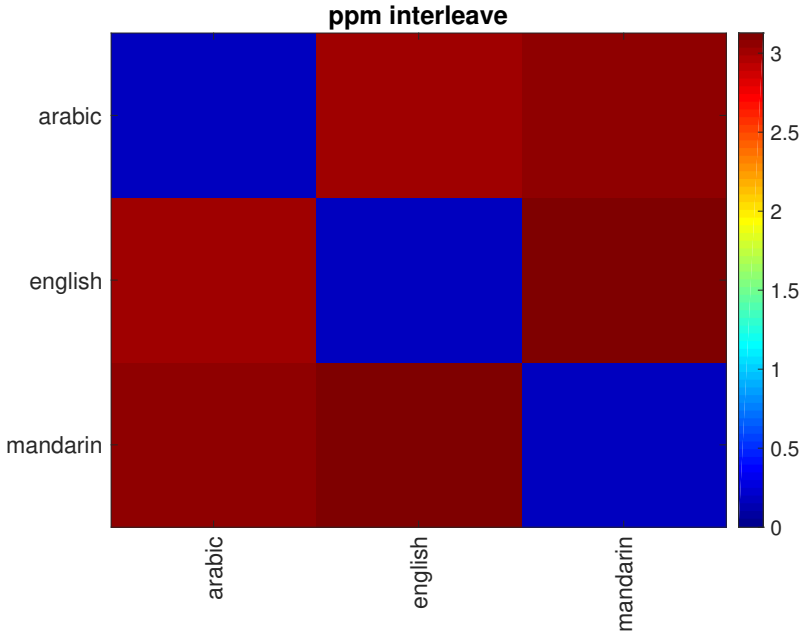


(b) with interleave

Figure 5.25: The 21 UNDHR video languages distances are computed by bzip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 5.24(a) shows the non-interleaved result and Figure 5.25(b) shows the interleaved result.

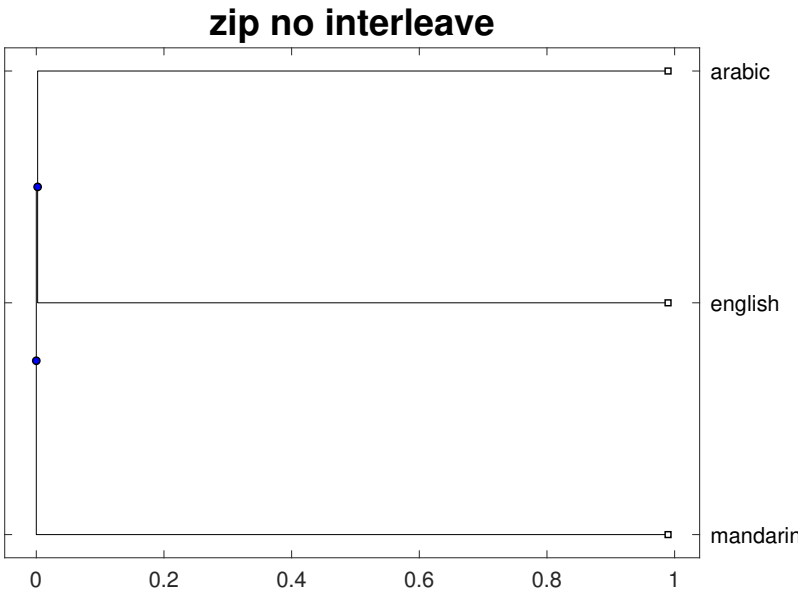


(a) without interleave



(b) with interleave

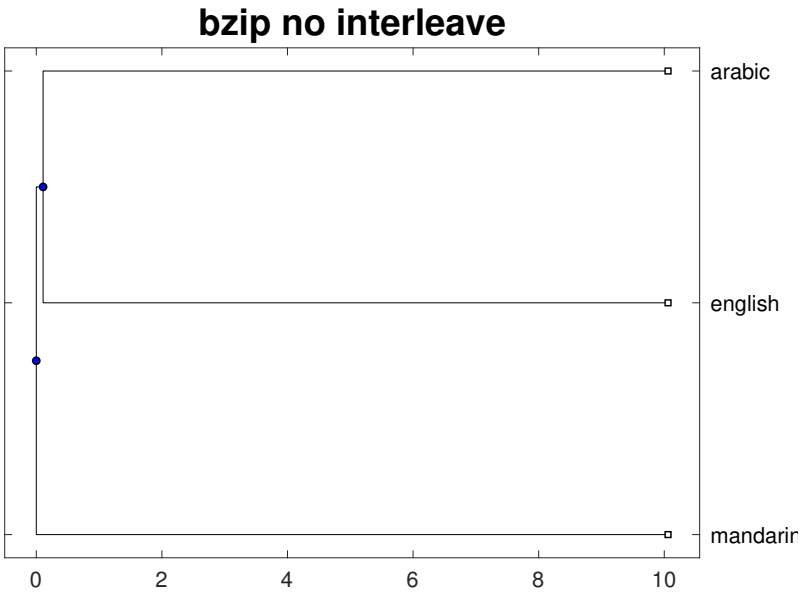
Figure 5.26: The 21 UNDHR video languages distances are computed by ppm and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 5.26(a) shows the non-interleaved result and Figure 5.26(b) shows the interleaved result.



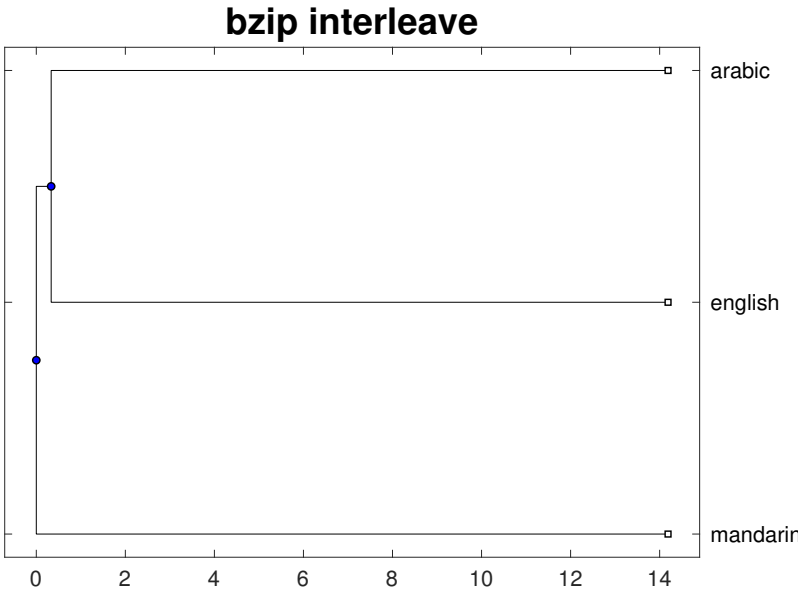
(a) without interleave

(b) with interleave

Figure 5.27: The 21 UNDHR video languages distances are computed by zip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 5.27(a) shows the non-interleaved result and Figure 5.27(b) shows the interleaved result.

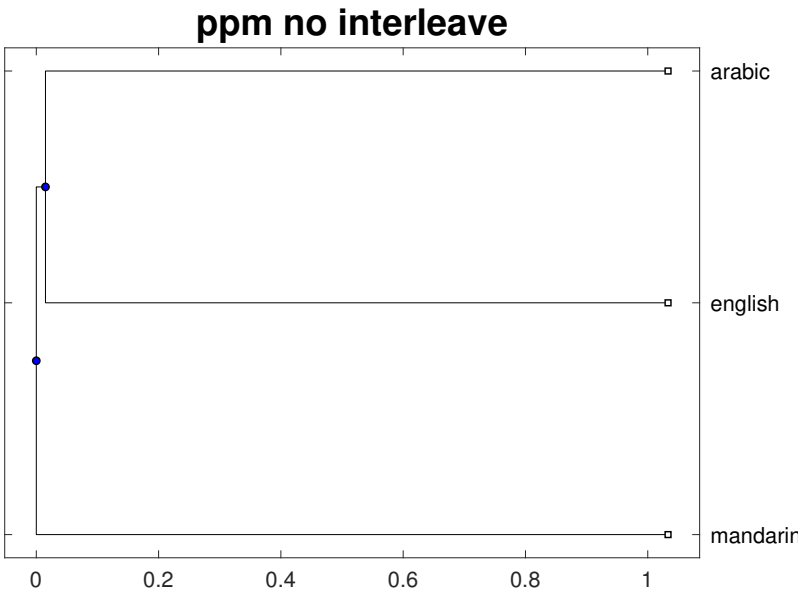


(a) without interleave

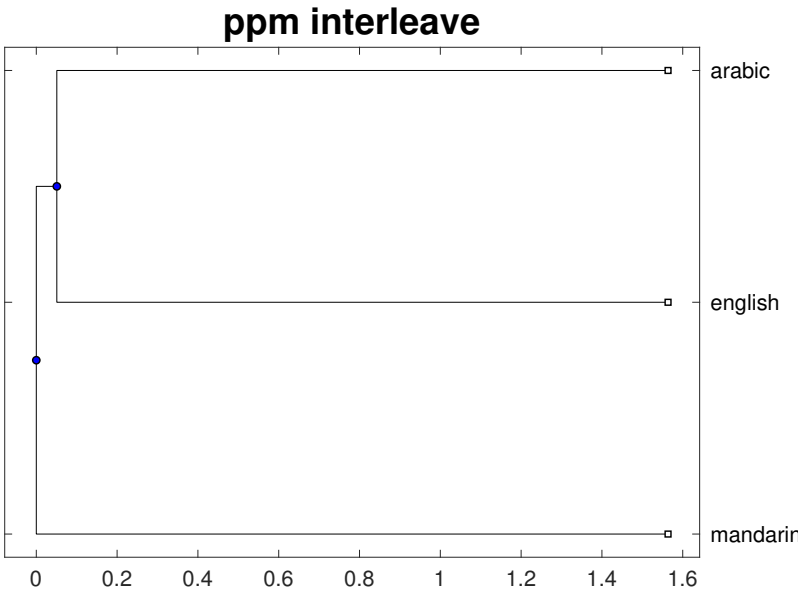


(b) with interleave

Figure 5.28: The 21 UNDHR video languages distances are computed by bzip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 5.28(a) shows the non-interleaved result and Figure 5.28(b) shows the interleaved result.



(a) without interleave



(b) with interleave

Figure 5.29: The 21 UNDHR video languages distances are computed by ppm and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 64. Figure 5.29(a) shows the non-interleaved result and Figure 5.29(b) shows the interleaved result.

5.3.5 Language distance results with 128 bins

This section describes the video language distances by using colour maps, phylogenetic trees and histogram distributions. The number of VQ bins is 128. The description of phylogenetic tree is in Section 3.2.2 and the description of histogram distribution is in Section 3.2.1.2. Figure 5.30 to 5.32 show the colour maps of the languages distances. Figure 5.33 to 5.35 show the dendrograms of language distances.

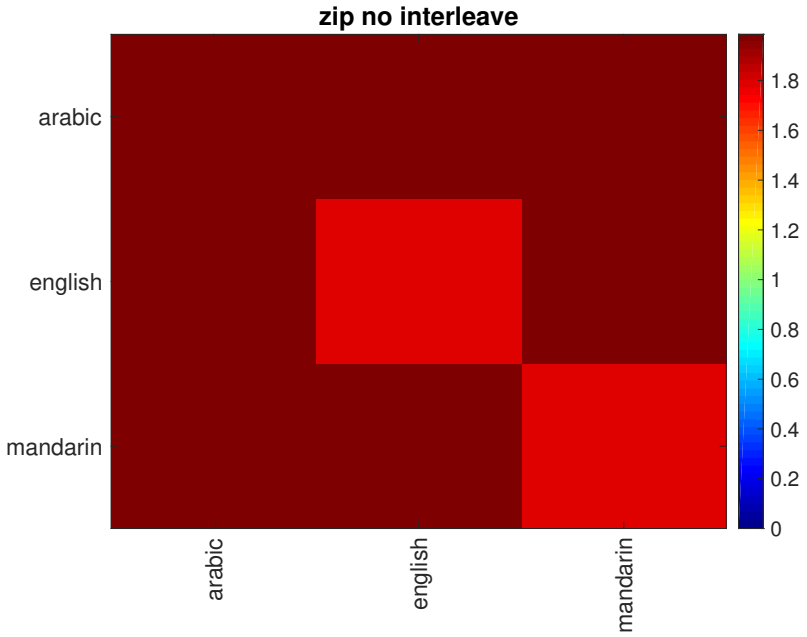
Table 5.9: Entropy(top) and accuracy(bottom) values with histogram binwidth = 1.93, vq binsize = 128.

	Zppm0	Zppm1	Zzip0	Zzip1	Zbzip0	Zbzip1
Entropy	1.00	0.65	0.92	1.00	1.00	0.00
Accuracy	1.00	1.00	1.00	1.00	1.00	1.00

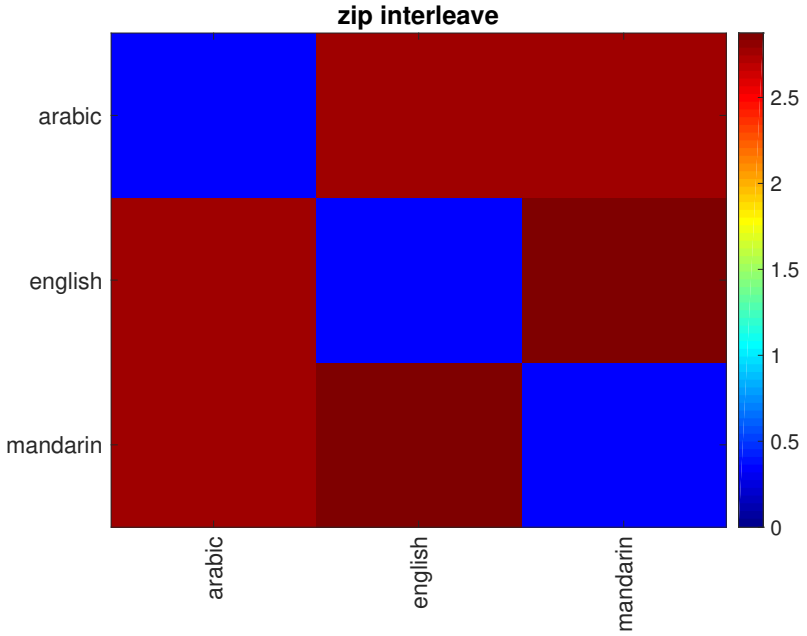
Table 5.9 concludes the entropy values of the histogram distributions for ppm, zip and bzip with interleaved and non-interleaved data. The results show the recognition accuracies of all compressions are 100% and the highest entropy is 1. For the same reason as 16 VQ bins, a histogram with two equiprobable bins would have an entropy of 1 bit. Thus 1.00 bits in this case still indicates a very non-smooth histogram (an all-or-nothing distance). And the Zbzip interleaving result is still 0 which means the language distances are very similar and there are no differences between them.

Figure 5.30 to Figure 5.35 show the colour maps and dendrograms of the pairwise language distances for each compression with interleaved and non-interleaved data. For zip, bzip and ppm, like previous sections, the interleaving result shows good performances of language identification while both the interleaving and the non-interleaving result can hardly show the distances relationships between the languages and the variation is much lower than Cavnar and Trenkle [1994]'s n-gram model. The ppm (Figure 5.32) and bzip (Figure 5.31) show the same problem as the zip results. Like the VLID n-gram results, we compare the VLID zipping trees with ALID ones under the same VQ binsize. For zip, bzip and ppm interleaving and non-interleaving result, the ALID results (Figure 4.35, 4.36, 4.37) show that

Arabic is close to Mandarin (except the ppm with interleaving method shows a closer distances between English and Arabic) while the VLID result in zip shows English is more closer to Arabic. Although ppm with interleaving method shows the same result as ALID, as we previously mentioned in TLID (Section 3.3.1.3), the interleaving method destroys the internal relationship between characters, the relationships between VLID feature are also impacted and the distances are related to compressibility instead of the relationships between the features. In that case, we think the 128 VQ bins case also performs poorly in VLID.

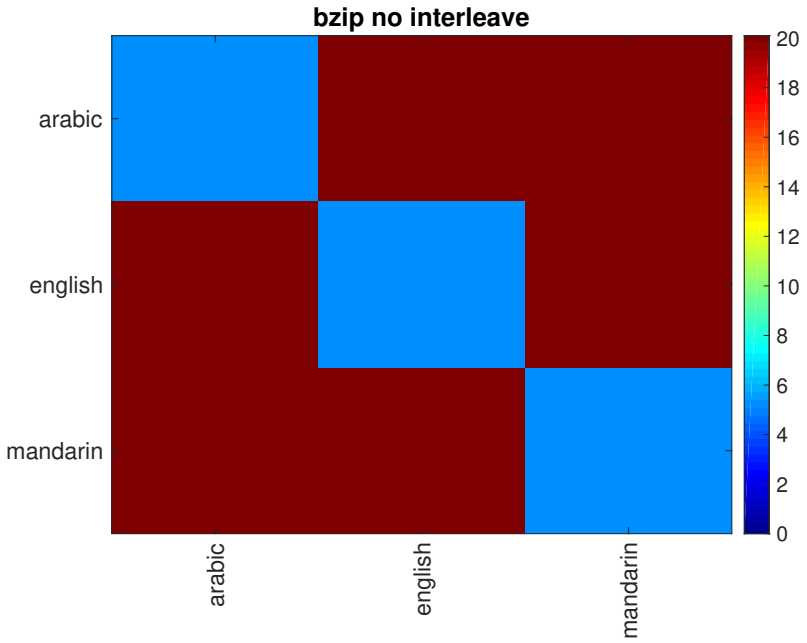


(a) without interleave

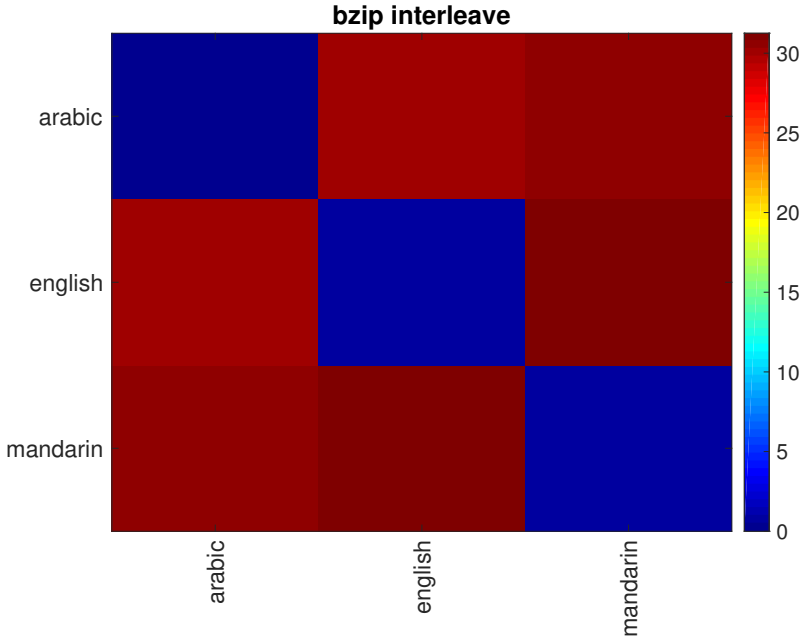


(b) with interleave

Figure 5.30: The 21 UNDHR video languages distances are computed by zip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 5.30(a) shows the non-interleaved result and Figure 5.30(b) shows the interleaved result.

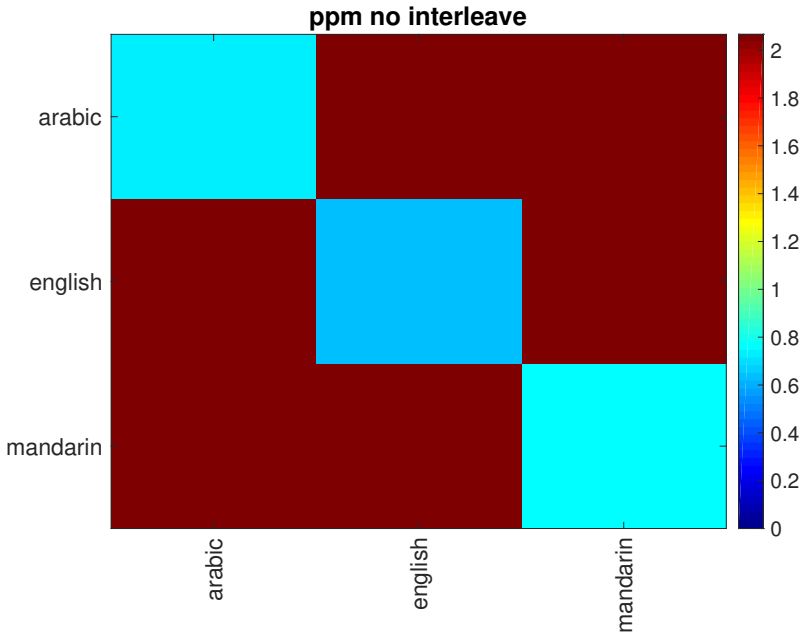


(a) without interleave

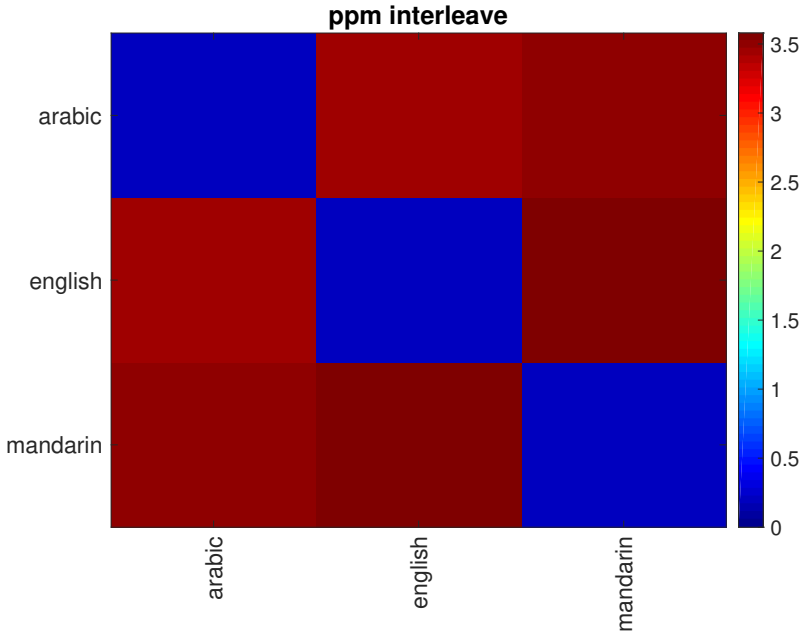


(b) with interleave

Figure 5.31: The 21 UNDHR video languages distances are computed by bzip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 5.30(a) shows the non-interleaved result and Figure 5.31(b) shows the interleaved result.

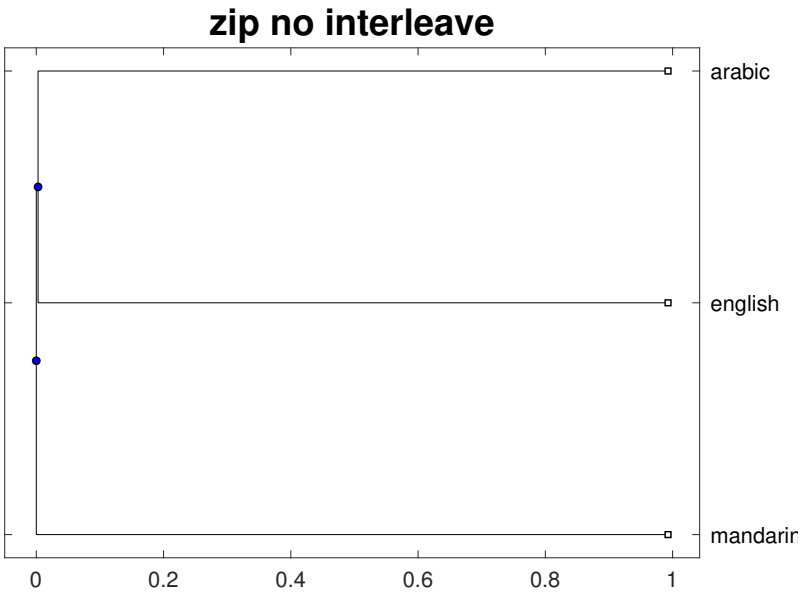


(a) without interleave



(b) with interleave

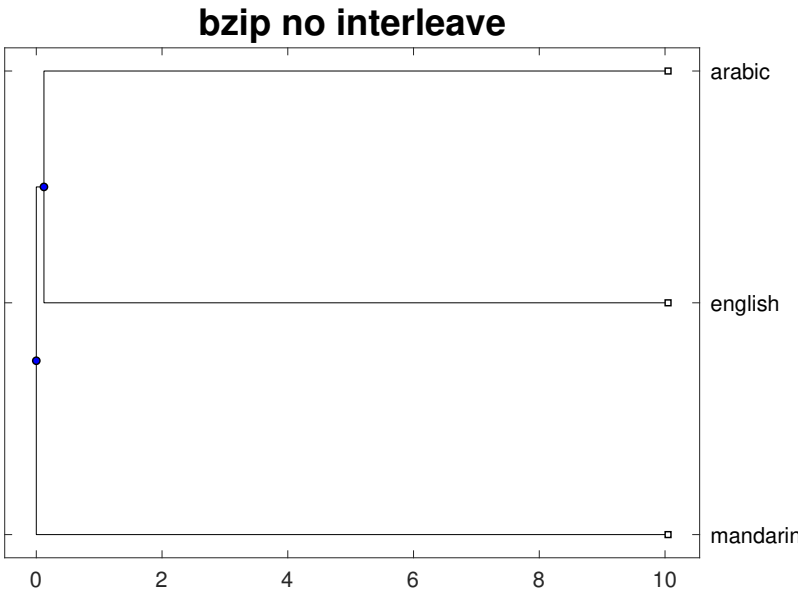
Figure 5.32: The 21 UNDHR video languages distances are computed by ppm and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 5.32(a) shows the non-interleaved result and Figure 5.32(b) shows the interleaved result.



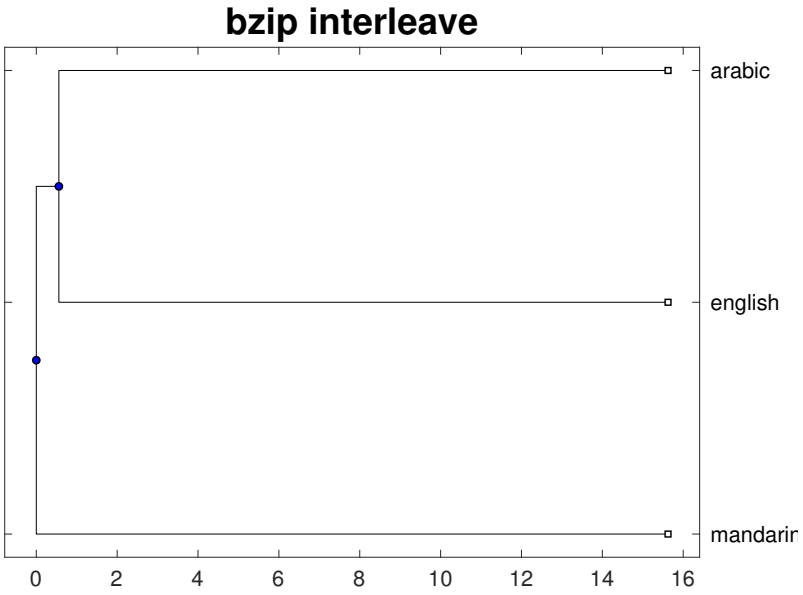
(a) without interleave

(b) with interleave

Figure 5.33: The 21 UNDHR video languages distances are computed by zip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 5.33(a) shows the non-interleaved result and Figure 5.33(b) shows the interleaved result.

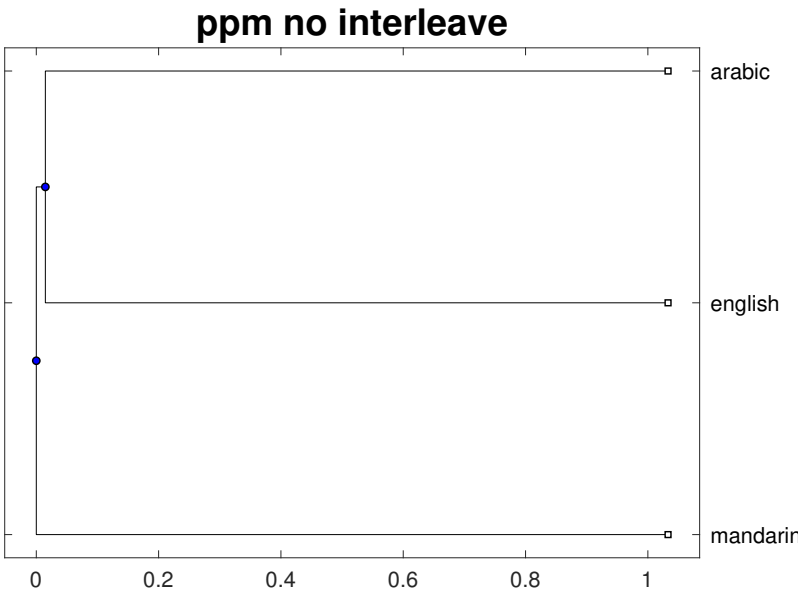


(a) without interleave

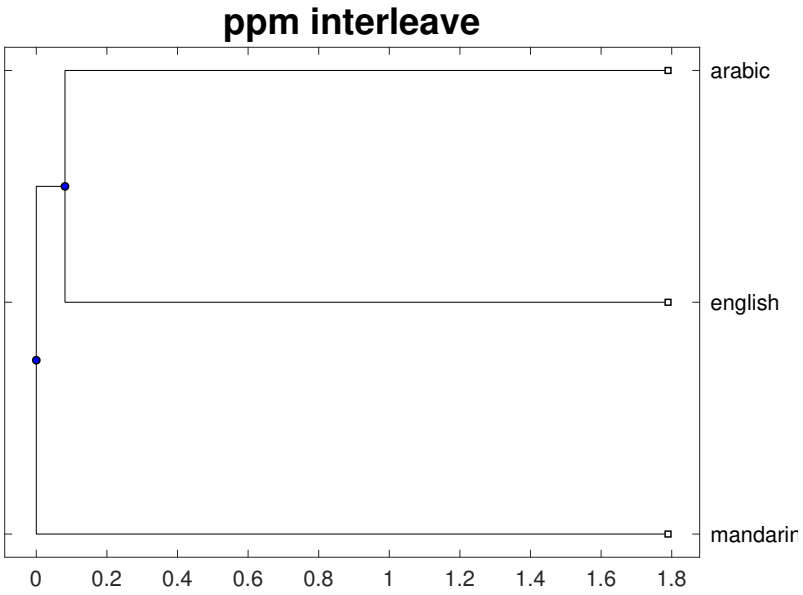


(b) with interleave

Figure 5.34: The 21 UNDHR video languages distances are computed by bzip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 5.34(a) shows the non-interleaved result and Figure 5.34(b) shows the interleaved result.



(a) without interleave



(b) with interleave

Figure 5.35: The 21 UNDHR video languages distances are computed by ppm and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 128. Figure 5.35(a) shows the non-interleaved result and Figure 5.35(b) shows the interleaved result.

5.3.6 Language distance results with 256 bins

This section describes the video language distances by using colour maps, phylogenetic trees and histogram distributions. The number of VQ bins is 256. The description of phylogenetic tree is in Section 3.2.2 and the description of histogram distribution is in Section 3.2.1.2. Figure 5.36 to 5.38 show the colour maps of the languages distances. Figure 5.39 to 5.41 show the dendrograms of language distances.

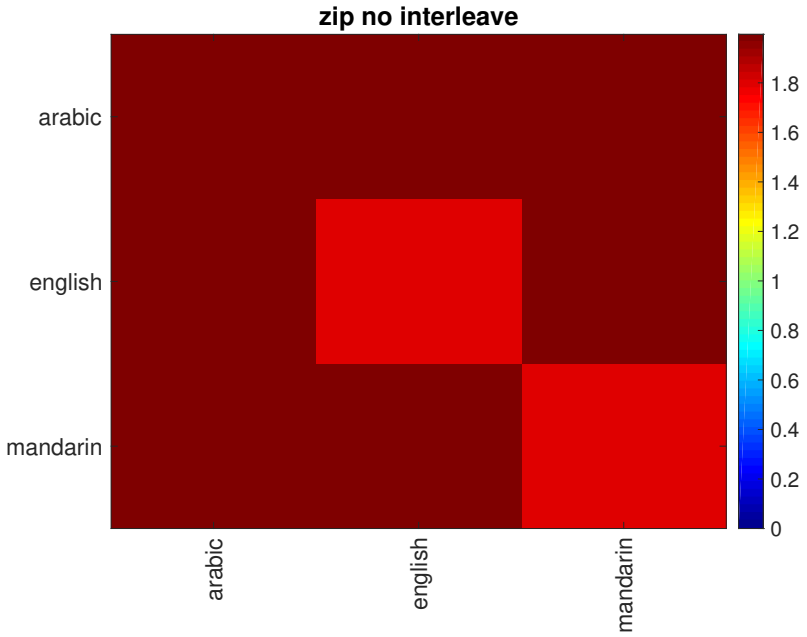
Table 5.10: Entropy(top) and accuracy(bottom) values with histogram binwidth = 1.93, vq binsize = 256.

	Zppm0	Zppm1	Zzip0	Zzip1	Zbzip0	Zbzip1
Entropy	1.00	0.92	0.92	1.00	1.00	0.00
Accuracy	1.00	1.00	1.00	1.00	1.00	1.00

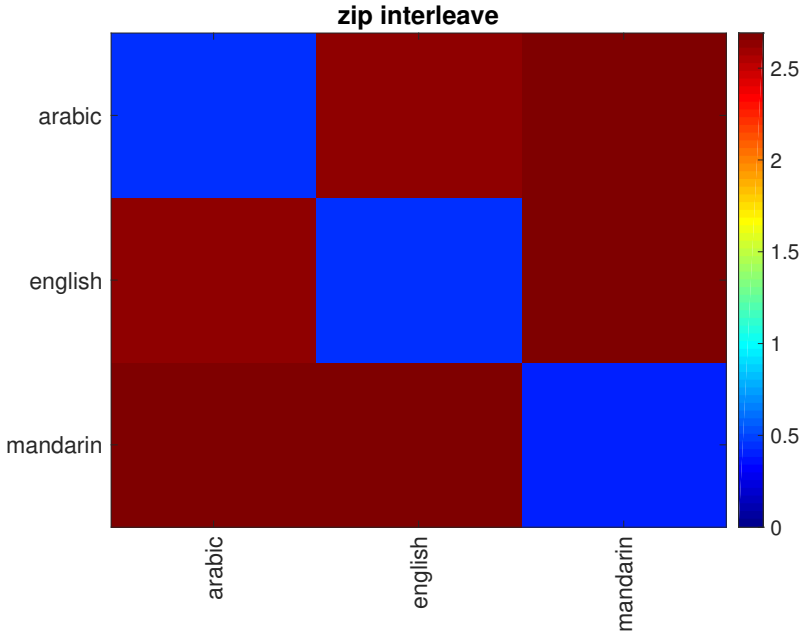
Table 5.10 concludes the entropy values of the histogram distributions for ppm, zip and bzip with interleaved and non-interleaved data. The results show the recognition accuracies of all compressions are 100% and the highest entropy is 1. For the same reason as 16 VQ bins, a histogram with two equiprobable bins would have an entropy of 1 bit. Thus 1 bit in this case still indicates a very non-smooth histogram (an all-or-nothing distance). And the Zbzip interleaving result is still 0 which means the language distances are very similar and there are no differences between them.

Figure 5.36 to Figure 5.41 show the colour maps and dendrograms of the pair-wise language distances for each compression with interleaved and non-interleaved data. For zip, bzip and ppm, like previous sections, the interleaving result shows good performances of language identification while both the interleaving and the non-interleaving result can hardly show the distances relationships between the languages and the variation is much lower than Cavnar and Trenkle [1994]'s n-gram model. The ppm (Figure 5.38) and bzip (Figure 5.37) show the same problem as the zip results. Like the VLID n-gram results, we compare the VLID zipping trees with ALID ones under the same VQ binsize. For zip, bzip and ppm interleaving and non-interleaving result, the ALID results (Figure 4.41, 4.42, 4.43) show that Mandarin close to Arabic

(except ppm with interleaving result whose Mandarin is close to English) while the VLID result shows that Arabic is close to English in zip, Mandarin is close to Arabic in bzip and the Mandarin is close to Arabic in ppm without interleaving result but is close to Arabic in the ppm with interleaving result. We can conclude that, like ALID in 256 VQ bin case, the distances are more unpredictable in 256 VQ bins. This is because the large binsize might split similar AAM features into different clusters and change the rank of the n-gram occurrences. Thus, the 256 VQ bins case also performs poorly in VLID.

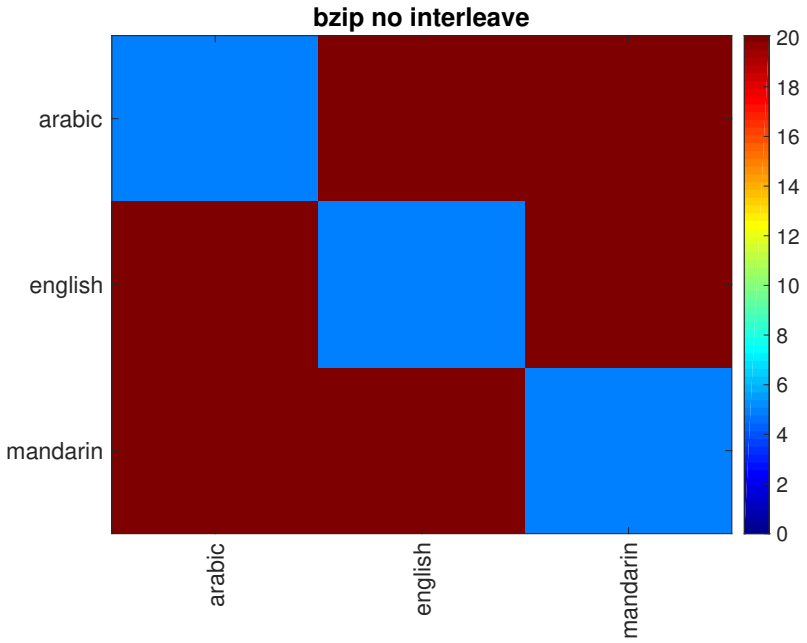


(a) without interleave

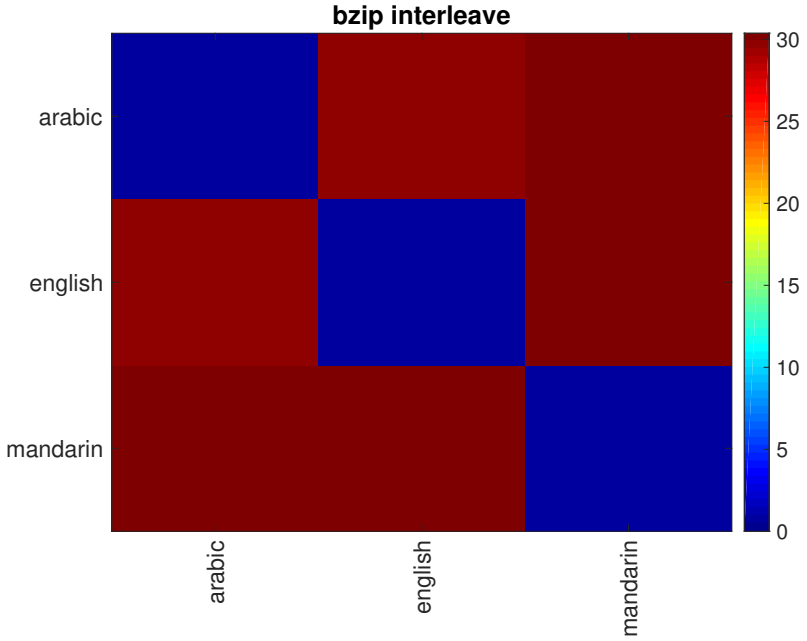


(b) with interleave

Figure 5.36: The 21 UNDHR video languages distances are computed by zip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 256. Figure 5.36(a) shows the non-interleaved result and Figure 5.36(b) shows the interleaved result.

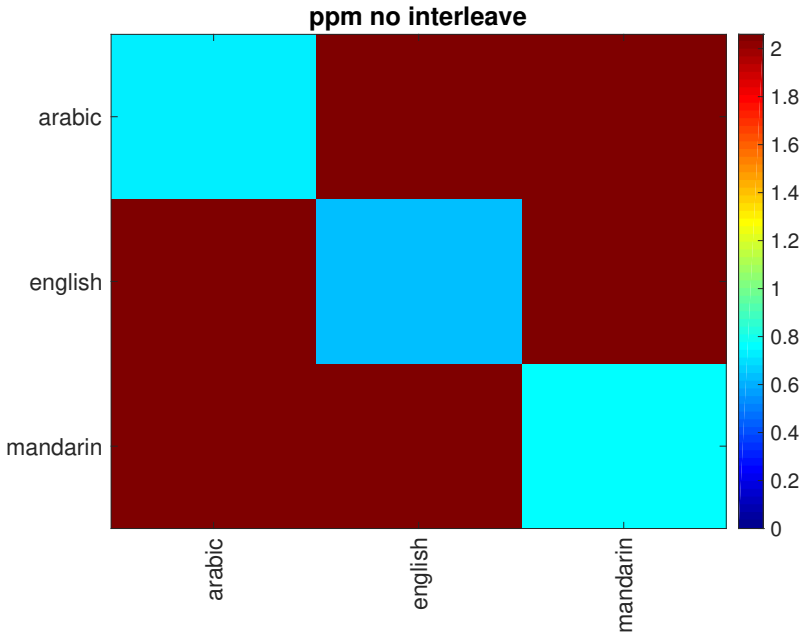


(a) without interleave

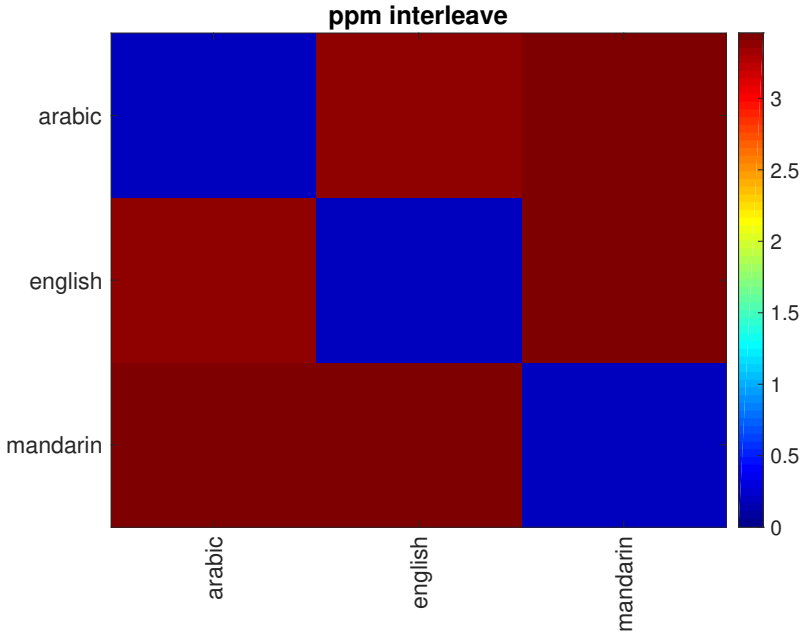


(b) with interleave

Figure 5.37: The 21 UNDHR video languages distances are computed by bzip and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 256. Figure 5.36(a) shows the non-interleaved result and Figure 5.37(b) shows the interleaved result.

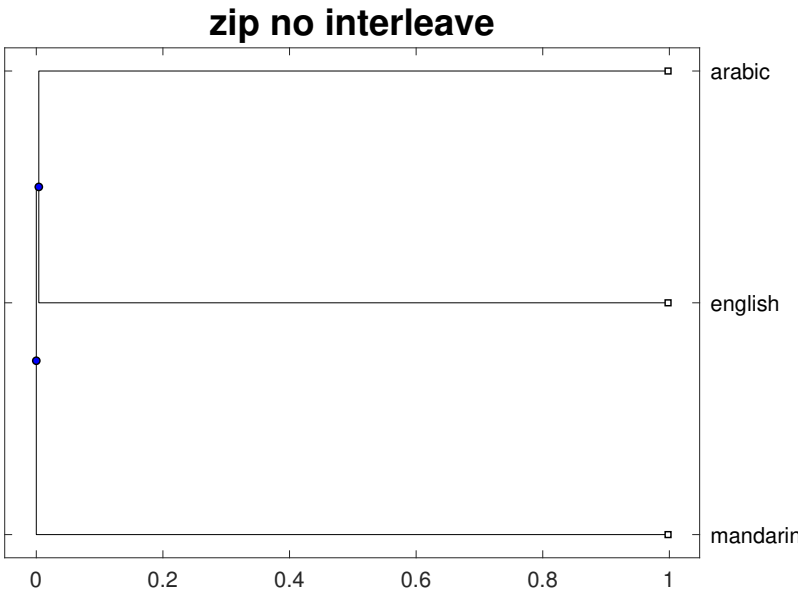


(a) without interleave

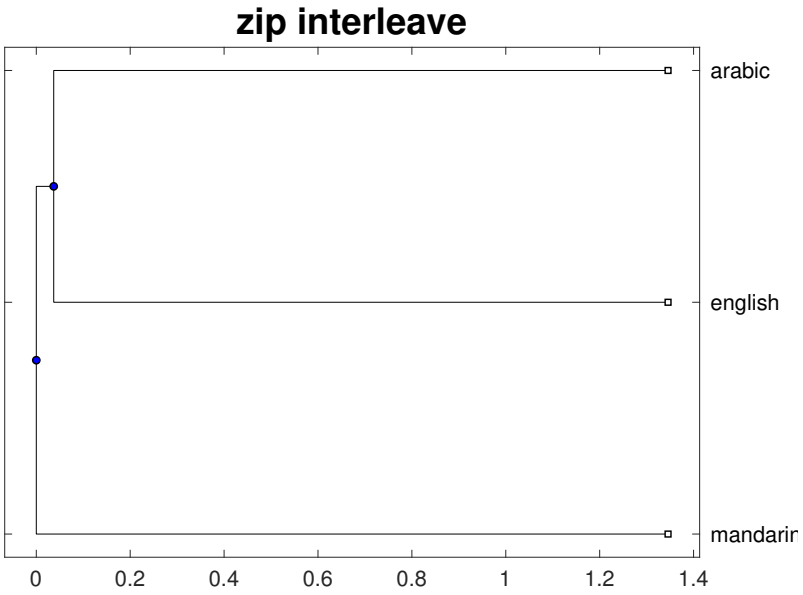


(b) with interleave

Figure 5.38: The 21 UNDHR video languages distances are computed by ppm and displayed by colour map. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 256. Figure 5.38(a) shows the non-interleaved result and Figure 5.38(b) shows the interleaved result.

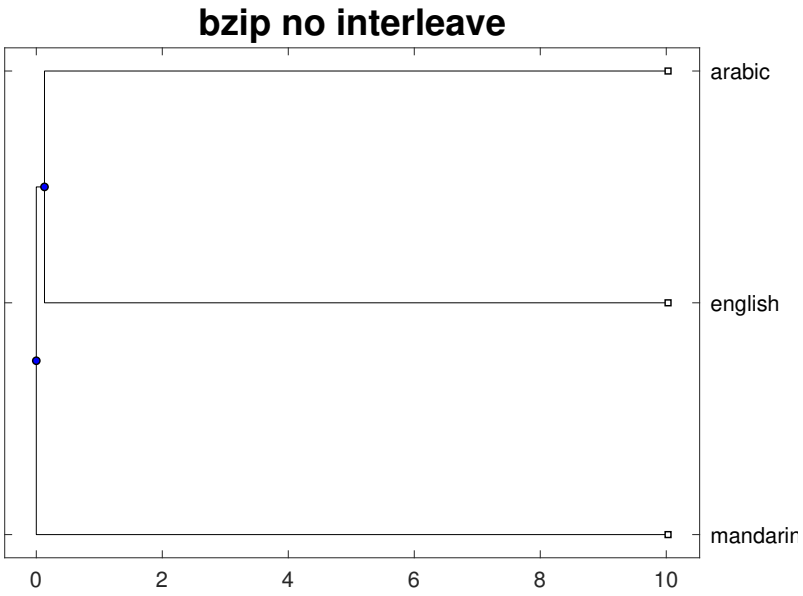


(a) without interleave

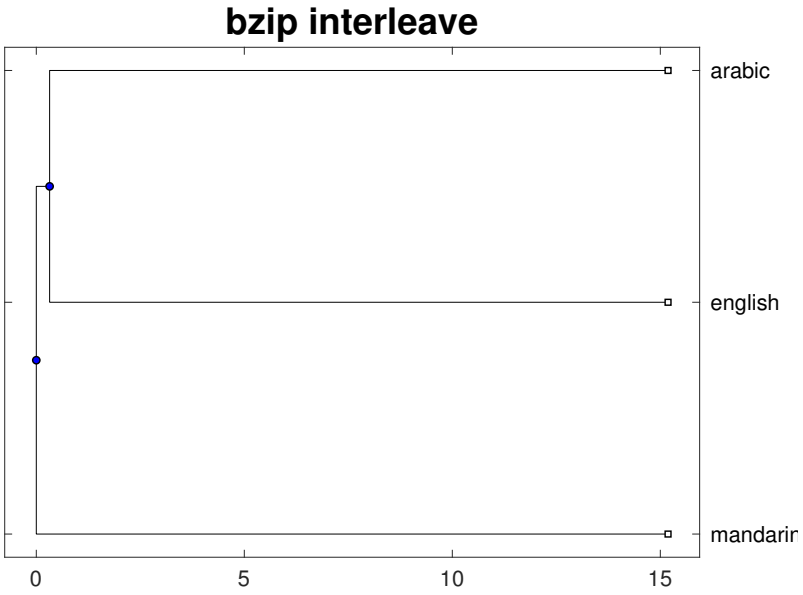


(b) with interleave

Figure 5.39: The 21 UNDHR video languages distances are computed by zip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 256. Figure 5.39(a) shows the non-interleaved result and Figure 5.39(b) shows the interleaved result.

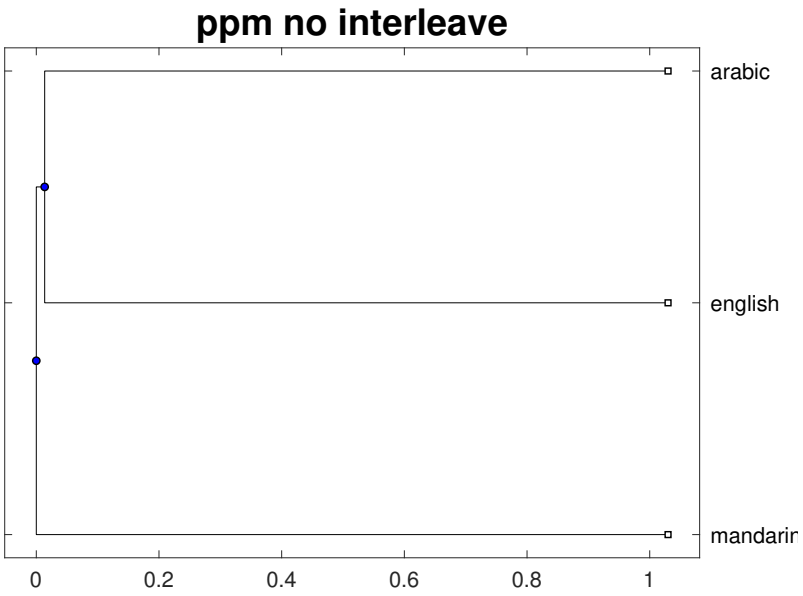


(a) without interleave

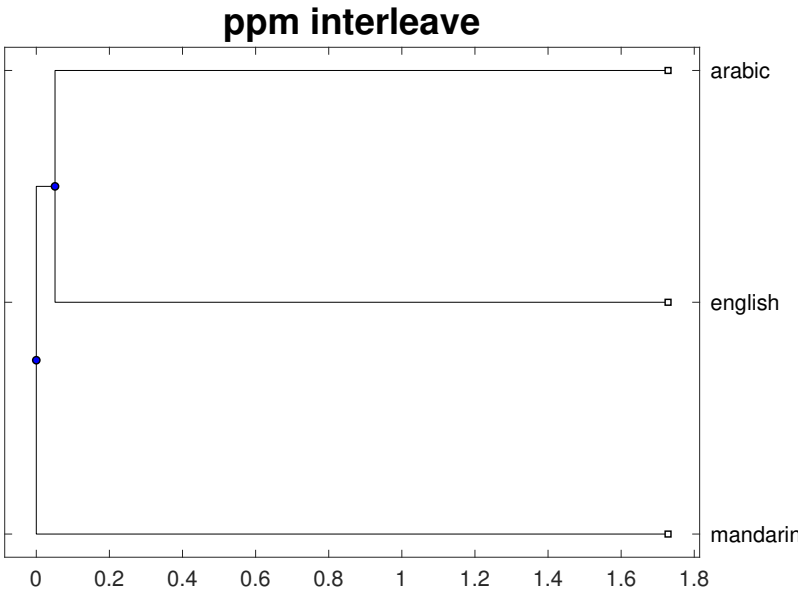


(b) with interleave

Figure 5.40: The 21 UNDHR video languages distances are computed by bzip and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 256. Figure 5.40(a) shows the non-interleaved result and Figure 5.40(b) shows the interleaved result.



(a) without interleave



(b) with interleave

Figure 5.41: The 21 UNDHR video languages distances are computed by ppm and displayed by dendrogram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The number of VQ bins is 256. Figure 5.41(a) shows the non-interleaved result and Figure 5.41(b) shows the interleaved result.

5.3.7 Conclusion

This section examines the zipping methods for video features. We can find the zipping is still an "all-or-nothing" method for TLID and the languages distances are too close to be discussed - one or two histogram bins are too spiky and do not make sense to calculate the entropies. We also compare the language distances with ALID results while all VLID language distances do not match with ALID ones. For lacking video data, it is hard to find the languages relationships by using zipping. And also, compared to the n-gram VLID result, the variation of the distances are still poor. We can conclude than zipping is probable for video language identification while it is not a good method to calculate the video language relationships.

5.4 Conclusion

In VLID, we apply the Cavnar and Trenkle [1994]'s n-gram model and zipping to see if we can find the relationships between the video languages.

The accuracy of VLID results for n-gram perform is low compare to TLID and ALID. Although we can use zipping to identify languages , it can not find the distances relationships, which is not probable to build language tree for comparison. For lack of data in VLID, the proof to describe the distances between English, Mandarin and Arabic is not enough. But we still can conclude that it is possible to use the Cavnar and Trenkle [1994]'s n-gram model for VLID and the n-gram model probably can show a higher recognition result with higher entropy compare to the current results.

Considering we have shown all TLID, ALID and VLID experiments, for the next chapter, we are going to investigate the relationships between those language distances.

Chapter 6

Tree Comparison and Mapping

6.1 Introduction

In previous chapters, we illustrate the relationships between languages by distance measurements formed in the text, audio and video domains. In Chapter 3, we measured the distances between text languages using Cavnar and Trenkle [1994]'s n-gram and zipping method. In Chapter 4, we measured the audio language distances via Cavnar and Trenkle [1994]'s n-gram, zipping and CK-distance method. In Chapter 5, we measured the video language distances via Cavnar and Trenkle [1994]'s n-gram and zipping.

In this chapter, we will discuss the use of the Robinson-Foulds and mapping results to describe if we can use the text language distances to find an unknown audio language. In this case, the data we used in this section is the language distances which are generated by the previous chapters. For TLID (Chapter 3), since both Cavnar and Trenkle [1994]'s n-gram model and zipping without interleaved data get the 100% accuracies with high entropy, we will use both of them in this task. For ALID, it shows a high accuracy in n-gram model with high entropy, and 100% accuracy in zipping with a lower entropy. Since the VLID results do not have enough evidences for the language relationships, we are not going to discuss it in

this chapter.

6.2 Language tree evaluation

To compare the language distance in TLID and ALID, the first question is, how close those language distances are. And we also wonder if the generated language trees, which based on the language features, are matched with the linguistic language tree.

To evaluate the results, we use the Robinson-Foulds tree distance measurement. The Robinson-Foulds is a tree distance measurement which is widely used in Phylogenetics and provides a linear computing time for rooted trees [Lu et al., 2017]. It also allows us to measure two rooted trees distances by branch partition without considering the branch length (the value of language distances).

6.2.1 Methods

To evaluate the language trees, we compare the TLID and ALID tree results with the background truth tree which is built by Ruhlen [1991]. As Ruhlen [1991] does not provide the language distances between Indo-Hittite and other subtrees, we only compare language trees based on the Indo-Hittite which contains most of the languages we use. We also create random trees 1000 times by using complete-linkage tree clustering (explained in Section 3.2.2) to see if the tree distances are better than the average of random trees with the ground truth tree. Table 6.1 shows the Indo-Hittite languages for building up the language tree.

Table 6.1: The languages which are used for Robinson-Foulds experiments.

Czech	Portuguese	Spanish	English	Polish
Italian	German	Swedish	Russian	

According to our previous conclusion, in TLID, the best result of n-gram is trigram with 100 penalty and the best performance of zipping is ppm without interleaving. For ALID, the Table 6.2 lists all best performance by VQ binsize for n-gram

and Table 6.3 lists the zipping result in ALID. Like previous chapters, for zipping results, a 0 following the name of the compressor denotes the interleaving status. For example, zip0, means the non-interleaved string with the zip compressor and ppm1 means an interleaved string with the ppm compressor.

We can find in ALID, the best performance (the highest accuracy and entropy) for n-gram model is the 32 VQ bins and the penalty is 100 with bigram. The best performance for zipping method, we remove the highest entropy in the 16 and the 256 VQ bins cases like the n-gram results show that both of them do not contain enough information for language identification. Thus, the best performance is ppm without interleaving in the 64 VQ bins.

Table 6.2: Summary of Cavnar and Trenkle [1994]'s n-gram results in ALID.

VQ binsize	Penalty	Gram	Accuracy	Entropy
16	50	2	0.76	2.87
32	100	2	0.86	2.78
64	10	1	0.84	2.84
128	400	5	0.8	2.88
256	50	1	0.78	2.83

Table 6.3: Summary of zipping results in ALID.

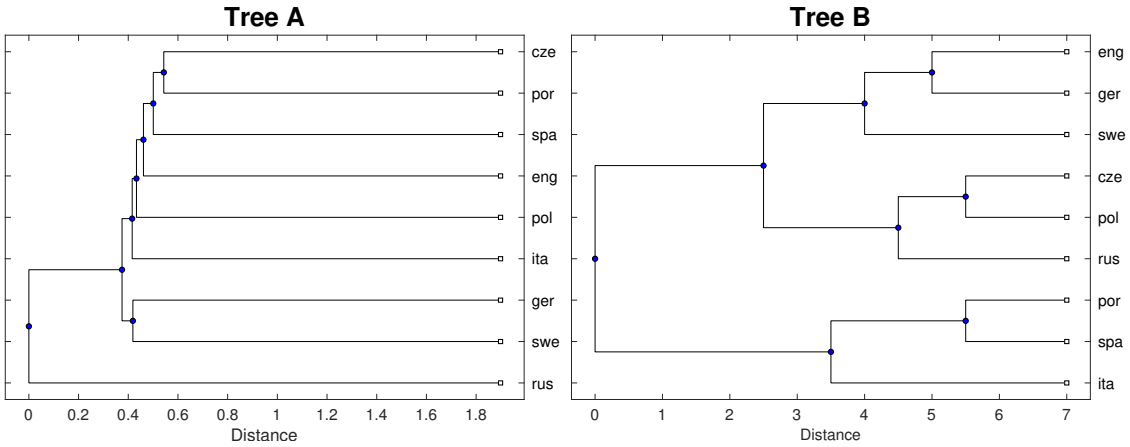
VQ binsize	Zipping	Accuracy	Entropy
16	ppm0	1	1.54
32	bzip0	1	1.06
64	ppm0	1	1.20
128	bzip0	1	1.02
256	zip0	1	1.33

6.2.2 Robinson-Foulds metric

As mentioned in Section 2.5.1, linguists define the generations and closeness between languages using a "language tree". In this thesis, we use different methods to measure the differences between languages and hence build language trees. Since a language tree may contain multiple languages in one node, it is not the same as a binary tree

distance measurement. Instead, we introduce the Robinson-Foulds metric[Robinson and Foulds, 1981] to measure the tree distances.

Robinson and Foulds [1981] stated that even the same data could be presented by different trees if using different methods. Before his study, most research focused on binary trees. To measure the tree that contains arbitrary nodes in one branch, the Robinson-Foulds metric separates the tree into several subsets by partition branches. The distance is calculated by the number of the sets in one tree that are not in other trees. These These distances are computed by considering all possible branches that could exist on the the two trees. Each branch divides the set of species into two groups



(a) Tree A (b) Tree B

Figure 6.1: An example of two language trees.

For example, Figure 6.1 shows two trees and both contain 9 languages: ita (Italian), pol (Polish), eng (English), spa (Spanish), cze (Czech), por (Portuguese), ger (German), swe (Swedish) and rus (Russian). To measure the distance between tree A and tree B, we separate tree A into subsets and transform it to the Newick tree format. So the tree A in Newick format is (rus, ((swe, ger), (ita, (pol, (eng, (spa, (por, cze)))))))) and the tree B is ((ita, (spa, por)),((rus,(pol, cze)), ((swe (ger, eng))))), which the parenthesis means the branchs in the tree. By cutting the tree branches, the subsets of these two trees are represented as S_A = (rus), (ger, swe, ita, pol, eng, spa, por, cze), (ita, pol, eng, spa, por, cze), (pol, eng, spa, por, cze),

(eng, spa, por, cze), (eng, spa, por, cze), (spa, por, cze), (por, cze) and B into set S_B = (ita, spa, por), (spa, por), (rus, pol, cze, swe, ger, eng), (pol, cze), (swe, ger, eng), (ger, eng). Comparing S_A and S_B, there is no subset appear in the both trees. Thus, by calculating the number of parenthesis, we can find that S_A contains 8 subsets and S_B contains 6 subsets, so the distance between tree A and tree B is $D_{AB} = 8 + 6 = 14$.

6.2.3 Results

This section explains the results of Robinson-Foulds tree comparison. According to the previous conclusions, the best performance in TLID n-gram is the TLID 3-gram with 100 penalty (See Figure 3.7(b)), the best performance in TLID zipping is the TLID ppm without interleaving (See Figure 3.21(b)), the best performance in ALID n-gram is the ALID 2-gram with 100 penalty in 32 VQ binsize (See Figure 4.7(b)) and the best performance in ALID zipping is the ALID PPM without interleave in 64 VQ binsize (See Figure 4.31(a)). And also in previous sections, we use the 10-fold cross validation to measure the language distances for each method. Thus, for each method, we can build 10 language trees. Here, we compare these trees with the linguistic language tree by using the Robinson-Foulds distance measurement. Table 6.4, 6.5, 6.6 and 6.7 show the Newick format of these language trees and Table 6.8 shows the Robinson-Foulds average distances of the linguistic language tree and the 10-fold cross validation results. We also build 1000 random trees by using random distances to see if the results are better than the random case.

We can find the linguistic language is close to itself and the random tree shows a lower distance compare to other trees. It means, although the generated language trees look different, the Robinson-Foulds method still view that they are the same distances to the linguistic language tree. What is more, the generated language trees do not perform better than the random tree. We can conclude that the generated language trees are different from the linguistic language tree.

Since the generated languages are not close to the linguistic tree, we wonder that

Table 6.4: List of the Newick format of the TLID n-gram language trees. The TLID n-gram tree is built based on the TLID 3-gram tree with 100 penalty result.

TLID 3-gram tree with 100 penalty
((english,(swedish,(italian,(portuguese,spanish)))),(german,(russian,(czech,polish))));
(((italian,(portuguese,spanish)),(swedish,(english,german))),(russian,(czech,polish)));
(((spanish,(portuguese,italian)),(english,(german,swedish))),(russian,(czech,polish)));
(russian,((czech,polish),((german,swedish),((portuguese,spanish),(english,italian)))));
((czech,polish),(russian,((italian,(portuguese,spanish)),(english,(german,swedish)))));
(((spanish,(portuguese,italian)),(czech,polish)),(russian,(german,(english,swedish))));
(((german,swedish),((portuguese,spanish),(english,italian))),(russian,(czech,polish)));
(((portuguese,spanish),(swedish,(english,italian))),((czech,polish),(german,russian)));
(((english,italian),(swedish,(portuguese,spanish))),(russian,(german,(czech,polish))));
(((italian,(portuguese,spanish)),(german,(english,swedish))),(russian,(czech,polish)));

Table 6.5: List of the Newick format of the TLID zipping language trees. The TLID zipping tree is built based on the PPM without interleaving result.

PPM without interleaving
((english,(swedish,(italian,(portuguese,spanish)))),(german,(russian,(czech,polish))));
(((italian,(portuguese,spanish)),(swedish,(english,german))),(russian,(czech,polish)));
(((spanish,(portuguese,italian)),(english,(german,swedish))),(russian,(czech,polish)));
(russian,((czech,polish),((german,swedish),((portuguese,spanish),(english,italian)))));
((czech,polish),(russian,((italian,(portuguese,spanish)),(english,(german,swedish)))));
(((spanish,(portuguese,italian)),(czech,polish)),(russian,(german,(english,swedish))));
(((german,swedish),((portuguese,spanish),(english,italian))),(russian,(czech,polish)));
(((portuguese,spanish),(swedish,(english,italian))),((czech,polish),(german,russian)));
(((english,italian),(swedish,(portuguese,spanish))),(russian,(german,(czech,polish))));
(((italian,(portuguese,spanish)),(german,(english,swedish))),(russian,(czech,polish)));

if it is possible to compare the TLID language trees with ALID ones. For example, is the TLID n-gram tree close to the ALID one? So, to compare the TLID results with ALID ones, we use the Robinson-Foulds to compare the languages trees and we also compare the TLID trees with the random tree. Table 6.9 shows the Robinson-Foulds tree distances between the n-gram TLID and the n-gram ALID. We can find, unlike comparing to the linguistic tree, the average distance between the TLID n-gram tree and the ALID n-gram tree is closer than the random tree. Table 6.10 shows the Robinson-Foulds distance proportion of the randomly generated trees. There

Table 6.6: List of the Newick format of the ALID n-gram language trees. The ALID n-gram tree is built based on the ALID 2-gram with 100 penalty in 32 VQ binsize.

ALID 2-gram with 100 penalty in 32 VQ binsize
((russian,(english,italian)),((spanish,swedish),((portuguese,polish),(czech,german))));
((portuguese,(spanish,(swedish,(german,(czech,polish))))),(italian,(english,russian)));
((portuguese,(italian,(english,polish))),((spanish,(czech,russian)),(german,swedish)));
((russian,(english,italian)),((portuguese,polish),((czech,swedish),(spanish,german))));
((spanish,(swedish,(german,(czech,polish)))),(portuguese,(italian,(english,russian))));
((russian,(english,italian)),((spanish,polish),(portuguese,(swedish,(czech,german)))));
((swedish,((spanish,polish),(czech,german))),((english,italian),(portuguese,russian)));
((russian,(english,italian)),(swedish,(portuguese,(german,(spanish,(czech,polish))))));
(((swedish,(german,(czech,polish))),(portuguese,spanish)),(english,(italian,russian)));
((italian,(english,russian)),((swedish,(german,(czech,polish))),(portuguese,spanish)));

Table 6.7: List of the Newick format of the ALID zipping language trees. The ALID zipping tree is built based on the PPM without interleaving in 64 VQ binsize result.

PPM without interleaving in 64 VQ binsize
((czech,(german,swedish)),((english,italian),((portuguese,russian),(spanish,polish))));
(((portuguese,spanish),(czech,(german,swedish))),((english,italian),(polish,russian)));
((russian,(english,italian)),(polish,(spanish,((czech,swedish),(portuguese,german)))));
((spanish,(czech,(german,swedish))),(polish,((english,italian),(portuguese,russian))));
((italian,(russian,(portuguese,english))),((german,(czech,swedish)),(spanish,polish)));
(((polish,(portuguese,spanish)),(german,(czech,swedish))),(russian,(english,italian)));
((russian,(english,italian)),(polish,((portuguese,spanish),(czech,(german,swedish)))));
((russian,(english,italian)),(polish,((czech,portuguese),(spanish,(german,swedish)))));
((italian,(english,russian)),(polish,((czech,(portuguese,swedish)),(spanish,german))));
((italian,(english,russian)),(polish,((portuguese,spanish),(german,(czech,swedish)))));

is no random tree the same as the n-gram tree and most of the random trees have high distances.And by testing the probability of the null hypothesis of the Robinson-Foulds distances between the TLID trigram language trees and the ALID 2-gram trees and the random trees, the $p-$value of the $t-$test is $5.0472e-164 < 0.01$ which rejects the null hypothesis H_0 that there is no difference between the means. Thus, the distances between the TLID n-gram trees and the ALID n-gram trees are not generated by chance.

Table 6.11 shows the Robinson-Foulds distance between the TLID ppm without

Table 6.8: Robinson-Foulds average distances of the linguistic language tree and the TLID and the ALID results. Each method has 10 language trees which corresponds to the 10-fold cross validation results. The TLID n-gram tree is built based on the TLID 3-gram tree with 100 penalty result. The TLID zipping tree is built based on the PPM without interleaving result. The ALID n-gram tree is built based on the ALID 2-gram with 100 penalty and 32 binsize result. The ALID zipping tree is built based on the ALID PPM without interleaving and 64 VQ binsize result. The random tree result is the average distance of the 1000 random trees and the linguistic tree.

Method	Linguistic	Random tree	TLID 3-gram	TLID ppm
Linguistic	0	13.6	14	14
Method	ALID 2-gram	ALID ppm		
Linguistic	14	14		

interleaving and the ALID ppm without interleaving in 64 VQ binsize. Still, we can find the average distance between the TLID tree and the ALID tree is closer than the random tree. Table 6.12 shows the Robinson-Foulds distance proportion of the randomly generated trees. There is no random tree the same as the ppm trees and most of the random trees have high distances. And by testing the probability of the null hypothesis of the Robinson-Foulds distances between the TLID ppm language tree and the ALID ppm trees and the random trees, the $p-$value of the $t-$test is $1.9789e167 < 0.01$ which rejects the null hypothesis H_0 that there is no difference between the means. Thus, the distances between the TLID ppm trees and the ALID ppm trees are not generated by chance.

Table 6.9: Robinson-Foulds average distances of the n-gram TLID and the n-gram ALID. Each method has 10 language trees which corresponds to the 10-fold cross validation results. The TLID n-gram tree is built based on the TLID 3-gram tree with 100 penalty result. The ALID n-gram tree is built based on the ALID 2-gram with 100 penalty and 32 binsize result. The random tree result is the average distance of the random trees and the n-gram TLID tree.

Method	Random tree	ALID 2-gram	p-value
TLID 3-gram	13.75	12.12	$5.0472e-164$

Table 6.10: The proportion of the distances between the randomly generated trees and the TLID 3-gram trees.

Distance	8	10	12	14
Proportion	0.09%	1.43%	11.62%	86.86%

Table 6.11: Robinson-Foulds average distances of the TLID ppm trees without interleaving method and the ALID ppm tree without interleaving method. Each method has 10 language trees which corresponds to the 10-fold cross validation results. The TLID zipping tree is built based on the PPM without interleaving result. The ALID zipping tree is built based on the ALID PPM without interleaving and 64 VQ binsize result. The random tree result is the average distance of the random trees and the TLID ppm without interleaving tree.

Method	Random tree	ALID ppm0	p-value
TLID ppm0	13.73	12.2	$1.9789e-167$

Table 6.12: The proportion of the distances between the randomly generated trees and the ppm trees.

Distance	2	8	10	12	14
Proportion	0.01%	0.14%	1.07%	10.75%	88.03%

6.2.4 Conclusion

This section uses the Robinson-Foulds method on measuring the language trees for ALID and TLID results. We can find the linguistic language tree is far from the ALID and TLID language trees rather than the random trees. However, once we calculated Robinson-Foulds distances between the TLID trees and the ALID trees, the distances are better than the random trees. Thus, we assume that it is possible to compare the language trees from TLID to ALID but not the linguistic one. The reason that the linguistic language tree does not have a good performance might be that the linguists build the language tree not only on the language features, the other factors, like empirical classification and culture differences also impact on it. What is more, the language classification is not commonly agreed by linguists which makes the work much more harder. For example, the Japanese and Korean are classified under the Altaic language tree [Ruhlen, 1991] while Lee and Hasegawa

[2011] views Japanese and Korean as different. Comparing Table 6.9 and Table 6.11, the conclusion is, n-gram performs better than zipping. And also in previous chapters, the n-gram model shows a higher entropy of the distances distribution than zipping. So, we are going to use the n-gram results for Sammon mapping in the next section.

6.3 Sammon mapping with Shepard interpolation results

In previous section, we conclude that the n-gram performs a better language grouping and shows a better Robinson-Foulds tree distances from TLID to ALID. Now the question is, whether it is possible to map an unknown language from ALID to TLID and find what language is it close to? Since Sammon mapping can map to a variety of dimensions, is there a natural dimensionality in which the text points, for example, fit?

6.3.1 Methods

Figure 6.2 explains the idea of mapping from TLID to ALID. Suppose there is a large number of languages in the text dataset while only contains 3 languages in the audio dataset (these three audio languages L_1, L_2 and L_3 are also exist in the text dataset). The yellow dots mean the languages which are not in both datasets so they can not be used for mapping. Supposing there is an unknown audio language x, we can easily use the n-gram model to calculate the distances between the unknown language x with L_1, L_2 and L_3 in ALID. We expect if it is possible to find there is a known language x have similar distance relationships with L_1, L_2 and L_3 in the text dataset as the relationships in the audio dataset. As we already know what language it is in text, we can conclude that the unknown language x in the audio dataset is the known language x in the text dataset.

The basis of our mapping technique is to use an interpolation function based on distances. We apply the Sammon mapping with the Shepard interpolation function in this section which is explained in 6.3.2. The text language distances we use in this section is 3-gram with 100 penalty and the audio language distances we use is 2-gram with 100 penalty.

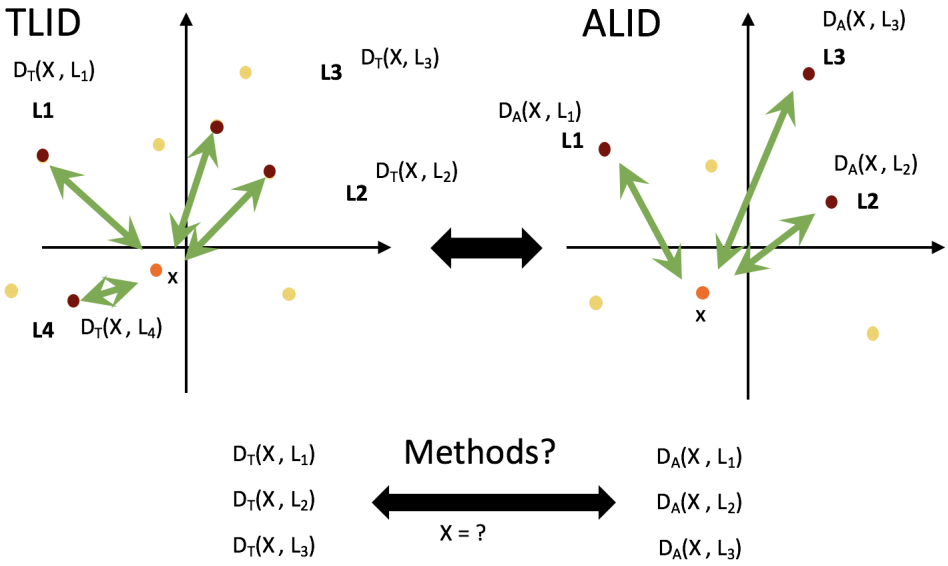


Figure 6.2: Explanation of Sammon mapping for language identification.

6.3.2 Sammon mapping

There are two main reasons why researchers reduce the dimensionality of a dataset. One is to simplify the dataset to save computing cost while preserving most of the relationship between the data, and the other is for better visualisation of the data structures.

Sammon mapping can preserve data structures with the minimum loss of information. One disadvantage of Sammon mapping is that as it calculates all interpoint distances, the complexity of mapping is very high and the computational speed is very slow. The other disadvantage of Sammon mapping is that it cannot process the unknown data [Pekalska et al., 1999]. It means that once there is unknown data coming into the dataset, all of the data must be mapped again and there is no guarantee that the surface of the mapped points will be the same as the previous mapped points. The Sammon mapping provides an idea about the mapping errors rate, which can be used to evaluate the mapping performance based on gradient descent. We use the algorithm designed and published by Cawley and Talbot[1]. The

[1]http://theoval.cmp.uea.ac.uk/matlab/default.html

method works according to the Sammon non-linear mapping algorithm.

Suppose the original matrix $\boldsymbol{A}$ in dimension $\boldsymbol{D}_1$ contains a number of vectors $\boldsymbol{v}_a, a \in 1...n$, $\boldsymbol{B}$ is the corresponding matrix of $\boldsymbol{A}$ in dimension $\boldsymbol{D}_2$ and the corresponding mapped vectors could be presented by $\boldsymbol{u}_a, a \in 1...n$. Thus, the pairwise Euclidean distance between $\boldsymbol{v}_a$ and $\boldsymbol{v}_b, a, b \in 1...n$ is s_{ab} and the pairwise Euclidean distance between $\boldsymbol{u}_{ab}, a, b \in 1...n$ is st_{ab}. So the loss information of the mapping can be calculated by the difference between the pairwise distance of the original and mapped matrices. Equation 6.1 shows the definition of the loss information e in dimension d at the t iteration[Sammon, 1969].

$$e(t) = \frac{1}{\sum_{a<b} s_{ab}} \sum_{a<b}^{n} \frac{(s_{ab} - st_{ab}(t))^2}{s_{ab}} \tag{6.1}$$

denoting y_{pq} is the $d \times n$ variables which is the mapped matrix in dimension d and corresponds to the error e, the the new mapped matrix at iterative time $t+1$ is

$$y_{pq}(t+1) = y_{pq}(t) - magicfactor \times \delta_{pq}(t) \tag{6.2}$$

where p is the length of the vectors and q is the dimension so $p = 1, ..., n$ and $q = 1, ..., d$. The magicfactor in Equation 6.2 is empirically to be 0.3 or 0.4 but the program we use replaces it by step-halving approach to make the algorithm works faster. The Euclidean distance of $st_{ab}(t)$ is $st_{ab}(t) = \sqrt{\sum_{m=1}^{d}(y_{am}(t) - y_{bm}(t))^2}$ and

$$\delta_{pq}(t) = \frac{\partial e(t)}{\partial y_{pq}} / |\frac{\partial^2 e(t)}{\partial y_p q(t)^2}| \tag{6.3}$$

The first derivative of e is

$$\frac{\partial e}{\partial y_{pq}} = \frac{-2}{\sum_{a<b} s_{ab}} \sum_{\substack{b=1 \\ j \neq p}}^{n} \left[\frac{s_{pb} - st_{pb}}{s_{pb} st_{pb}} \right] (y_{pq} - y_{bq}) \tag{6.4}$$

and the second derivative of e is worked out by the Hessian matrix which contains

all the second partial derivatives of e and is shown in Equation 6.5.

$$\frac{\partial^2 e}{\partial y_{pq}^2} = \frac{-2}{\sum_{a<b} s_{ab}} \sum_{\substack{b=1 \\ j \neq p}}^{n} \frac{1}{s_{pb} s t_{pb}} \left[(s_{pb} - s t_{pb}) - \frac{(y_{pq} - y_{bq})^2}{s t_{pb}} (1 + \frac{s_{pb} - s t_{pb}}{s t_{pb}}) \right] \quad (6.5)$$

6.3.3 Shepard's interpolation

Considering the fact that there are hundreds of text languages but only dozens of audio language datasets are currently available, it is necessary to find a proper method for comparison. As we suppose the distances between languages contain hidden relationships with each language, we expect the distances of the text languages to correspond reasonably closely to the audio language distances. Meanwhile, since the distances of text languages and audio languages are more likely to be an irregularly-spaced data issue, the method proposed by Shepard [1968] regarding a two-dimensional interpolation might be a possible option to solve this problem. Although Shepard's method is of limited help in describing the direction between the points, it is a simple and general method to implement and can show the language relationships in our project.

Shepard [1968]'s algorithm is a method that tries to explain the distance between points using simple and local functions that are called a weighted average of points. For example, supposing the data points $s_i, i \in (1...n)$ in dimension $\boldsymbol{D}_1$ could be interpolated to the same dimension $\boldsymbol{D}_2$ as the data points $q_i, i \in (1...n)$, thus each interpolated point $q_i = f(s_i)$ is a weighted average w_i of the values q_i.

The q_x, which is the interpolated value of s_x could be calculated by equation 6.6:

$$q_x = f(s_x) = \begin{cases} \frac{\sum_{i=1}^{n} w_i(s_x) q_i}{\sum_{i=1}^{n} w_i(s_x)}, & w \neq 0, i \in (1...n), \\ q_i, \quad w = 0, \quad i \in (1...n), \end{cases} \quad (6.6)$$

where w_i is

$$w_x = \frac{1}{dist(s_x, s_i)^p}, i \in (1...n) \tag{6.7}$$

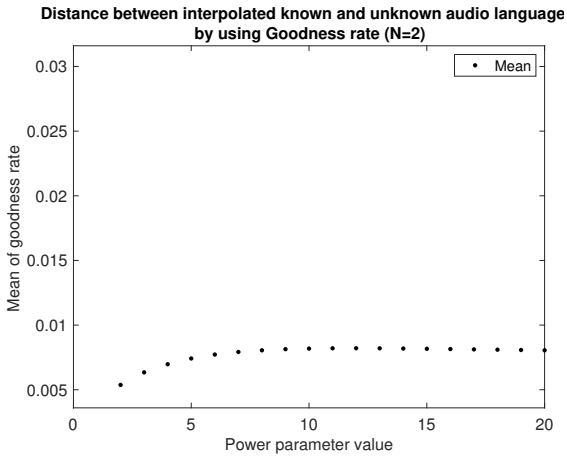
The weight p value in equation 6.6 decides whether the neighbouring points have a greater influence on interpolation than other points. If the value of p value increases, the greater the influence of the neighbour points.

6.3.4 Results

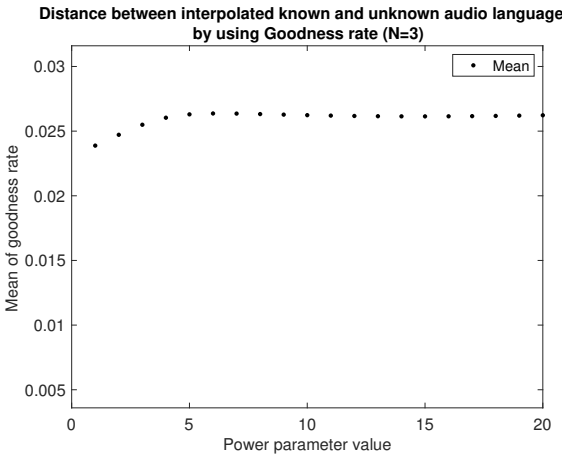
To measure the performance of the mapping results, we use a concept of "Goodness Ratio (GR)" to calculate the ratio of the language distances between itself and other mapped languages. In our ALID and TLID dataset, we have five languages that contain recordings of more than one speaker which is $C = \{Chinese, French, Javanese, Latin, Spanish\}$. We denote t_i is the TLID language points and a_j is the ALID language points which i is the number of TLID and j is the number of ALID language. Thus, the interpolated language points of TLID are tm_i and the interpolated language points of ALID are am_j. By interpolating am_j to TLID space, the interpolated points of am_j is as_j. We measure the Euclidean distances dm_{ij} between the as_j and the tm_i. So the Goodness ratio GR_{ij} of the mapping result can be calculated as:

$$GR_{ij} = \frac{\sum_{a \in C, b \in C} dm_{ab}}{\sum_{r \leq 1, q \leq 1}^{r=i, q=j} dm_{rq} - \sum_{a \in C, b \in C} dm_{ab}} \tag{6.8}$$

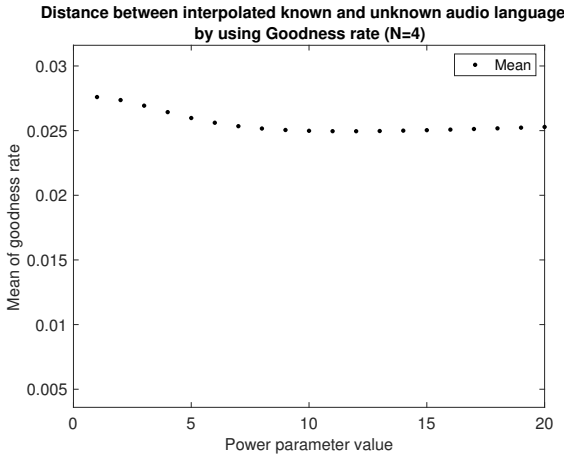
For Sammon mapping, we varied the dimension D from 2 to 20 and for Shepard's interpolation, the weight value p was evaluated from 1 to 20. We compared the GR in each pair of D and p. In each dimension, we measure the goodness ratio of the 5 languages distances with themselves versus the other language distances. The mean of the goodness ratio is the average of the five languages and the results are shown in Figure 6.3, 6.4, 6.5 and 6.6. We can conclude that the 2 dimension shows the lowest goodness ratio compare to the other dimensions.



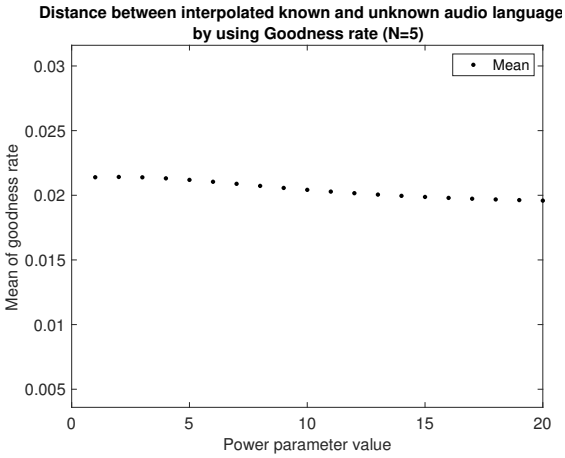
(a) $N = 2$



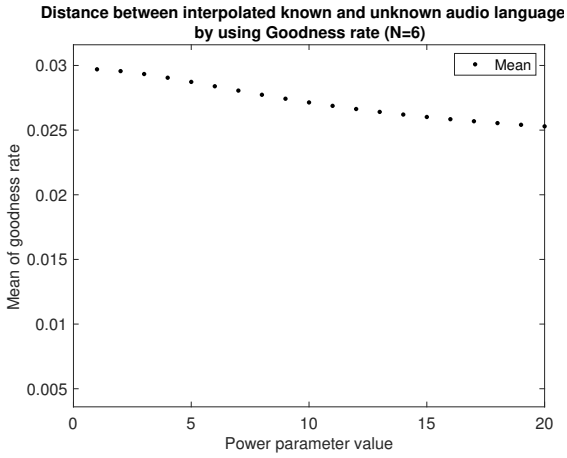
(b) $N = 3$



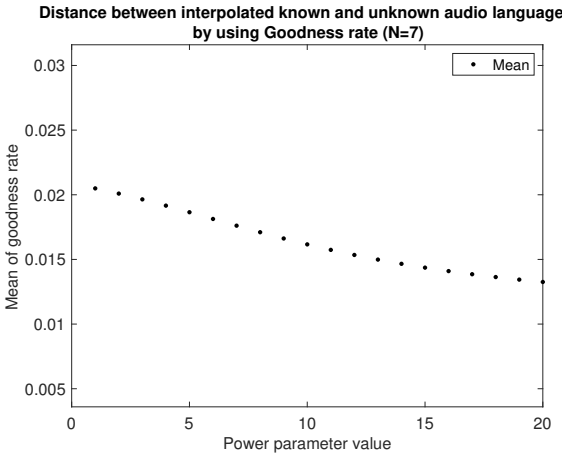
(c) $N = 4$



(d) $N = 5$

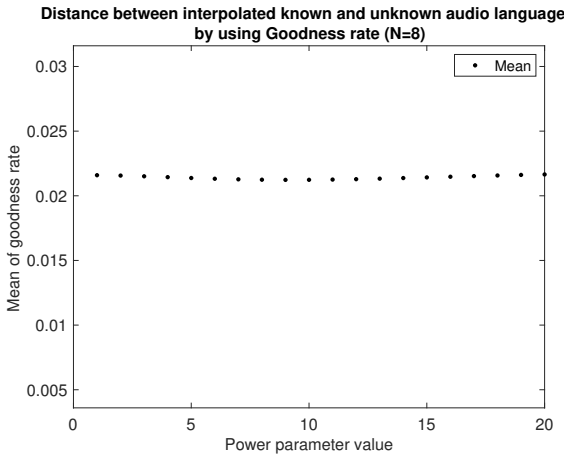


(e) $N = 6$

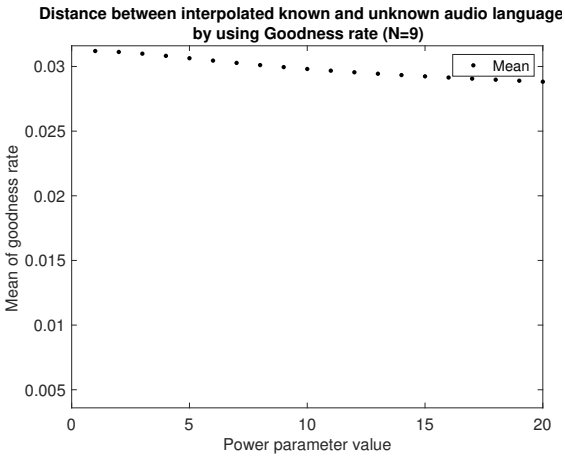


(f) $N = 7$

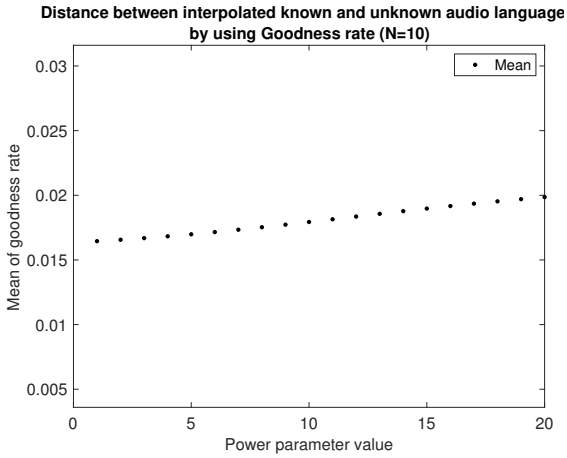
Figure 6.3: Goodness rate (Dimension $N = 2$ to 7)



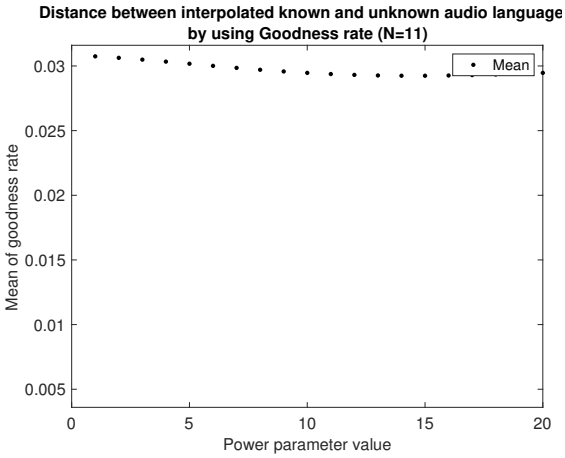
(a) $N = 8$



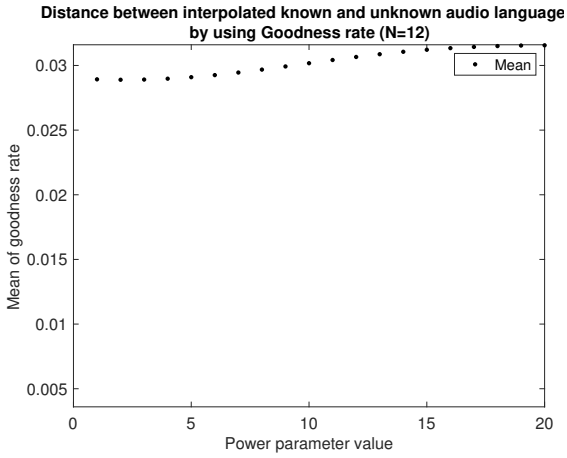
(b) $N = 9$



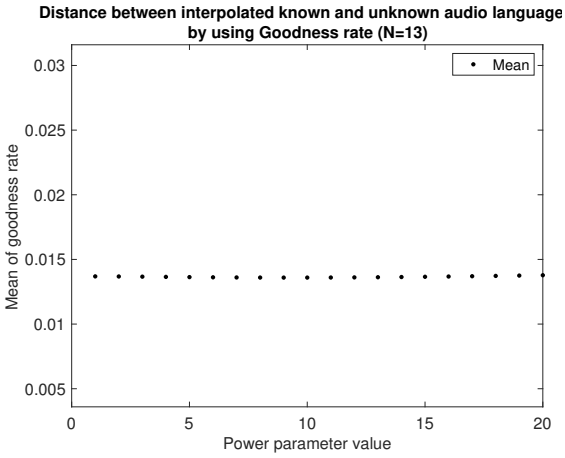
(c) $N = 10$



(d) $N = 11$

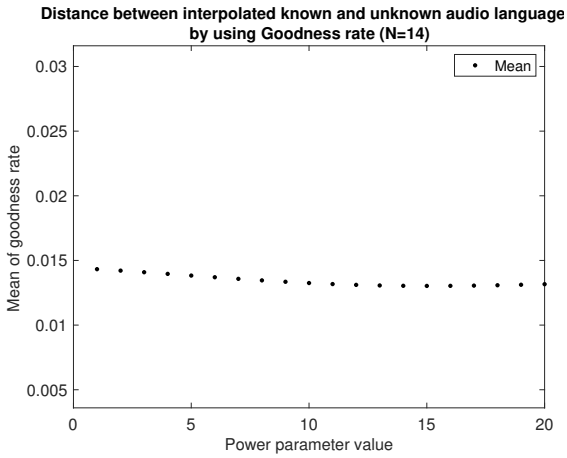


(e) $N = 12$

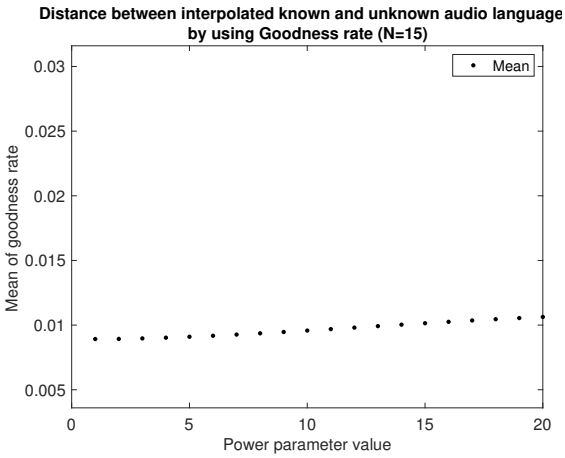


(f) $N = 13$

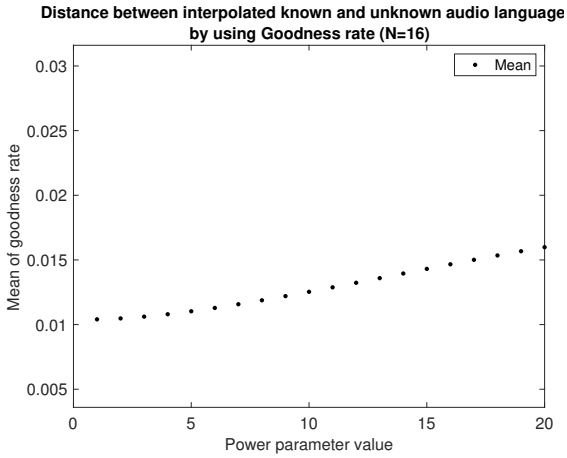
Figure 6.4: Goodness rate (Dimension $N = 8$ to 13)



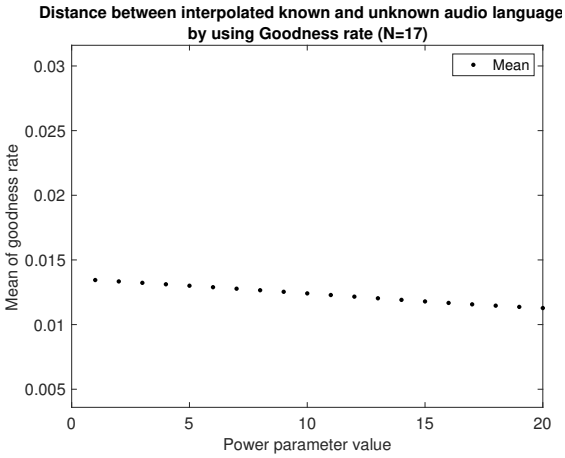
(a) $N = 14$



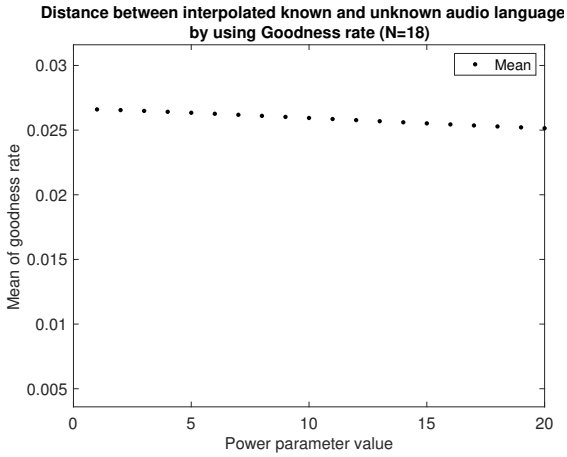
(b) $N = 15$



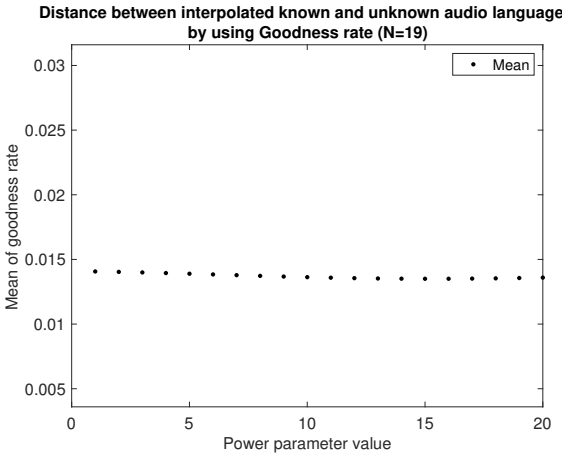
(c) $N = 16$



(d) $N = 17$

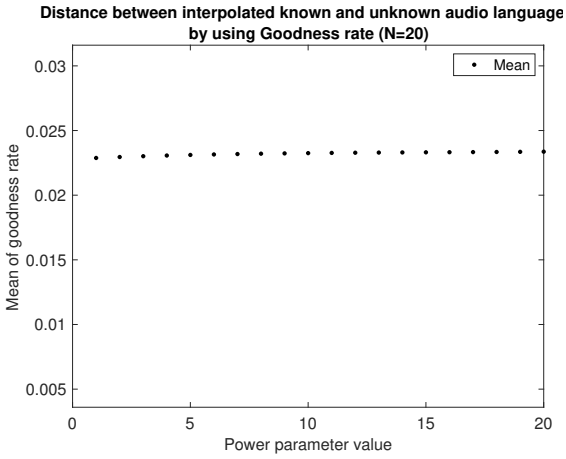


(e) $N = 18$



(f) $N = 19$

Figure 6.5: Goodness rate (Dimension $N = 14$ to 19)



(a) $N = 20$

Figure 6.6: Goodness rate (Dimension $N = 20$)

6.3.5 Conclusion

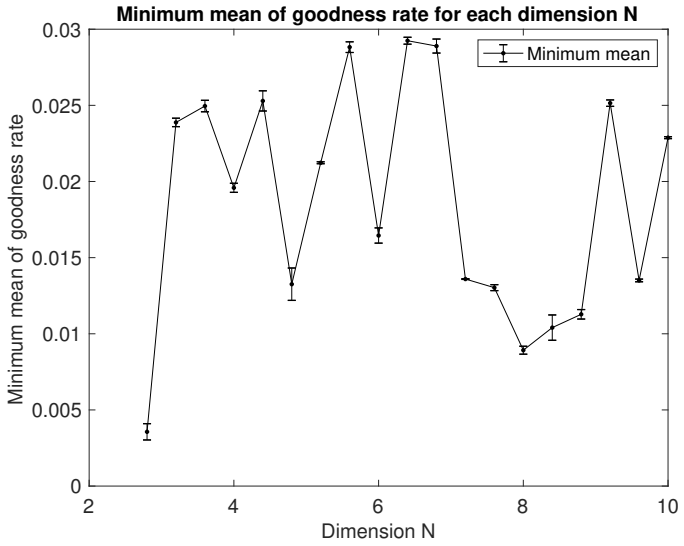


Figure 6.7: Minimum mean of Goodness for each dimension N

Figure 6.3 to Figure 6.6 is the mean variation of goodness ratio from the audio to the text dimension. 6.7 summarise the distribution of the minimum mean of GR for each dimension and the error bar is the mean ± 2 standard error.

We found that the lowest goodness ratio value is in dimension 2 with low standard error. This means the interpolation performs best at dimension 2. Thus, we can conclude that the optimal dimensionality for human language is 2.

Chapter 7

Conclusion and future work

7.1 Conclusion

In this thesis, we examine the TLID, ALID and VLID language distances based on several methods. As what we expect, we wonder if we can identify an unknown ALID language based on the TLID since text dataset is easy for collection and sufficiently provided on the Internet.

For TLID, we compare the language distances based on the Cavnar and Trenkle [1994]'s n-gram model and the zipping methods - zip, bzip and ppm. Cavnar and Trenkle [1994]'s n-gram is a high accuracy method which has been proved to work for language classification. Language classification is another method proposed by Benedetto et al. [2002]. The advantage of zipping is that it is parameter free to identify languages based on the compressibility entropy. What we have done is that, we introduce the language tree for n-gram model and evaluate the result by difference penalty and perform the distance results by using the histogram and entropy. We can find the Cavnar and Trenkle [1994]'s n-gram model shows a good language grouping in TLID and also the zipping without interleaving method performs well.

For ALID, to fit for TLID result, we introduce the Cavnar and Trenkle [1994]'s n-gram model and the zipping methods to find the language relationships. We

compare the accuracy and entropy for each penalty and VQ binsize and also build the language tree for each result to compare with the linguistic language tree. And also, we apply another method called CK-distance to measure the language distances by using the MPEG. We can find the n-gram model still perform a good accuracy with high entropy while the zipping methods show "all-or-nothing" result which has low entropy. As another kind of compression, CK-distance also shows the same problem as bzip, zip and ppm. So we conclude that the Cavnar and Trenkle [1994]'s n-gram model performs the best in ALID.

For VLID, the n-gram has a lower language identification accuracy compare to TLID and ALID, while it still performs a higher entropy compare to zipping. In VLID, we can conclude that zipping is an "all-or-nothing" method which is not appropriate for ALID and TLID. The generated language trees do not describe a lot of language relationships due to the lack of data but we assume it is possible to get a more accuracy language tree if there is enough data for VLID.

To identify the unknown audio language by using the text bases, we firstly introduce the Robinson-Foulds to comparing the language trees. We compare the generated language trees with the linguistic language tree. And also, we apply a random tree to see if the Robinson-Foulds distances between the TLID and ALID language trees are better than the random performance. The conclusion is that we find the TLID and ALID language trees are more similar rather than the linguistic language tree and the random tree. We suppose this is impacted by multiple factors such as the cultural and the empirical understanding about the languages. For example, linguists view Japanese and Chinese as two different languages, however, the text of Japanese contains a lot of Chinese characters which are called "Kanji". This is because Japanese learn Chinese characters during the Tang Dynasty.

As we conclude the best performance for our task is n-gram model, we introduce the Sammon mapping and the Shepard interpolation for mapping ALID to TLID. We use an evaluation method which is called "Goodness ratio" to see if the mapped languages are close to itself and we find the best performance is dimension 2.

7.2 Future work

In search of our methods, we obtain a lot of results from zipping, n-gram and Shepard's interpolation. However, it is unclear why the linguistic language tree is different from the ALID and TLID tree. One possible reason is that the language relationships are not fully investigated by linguists and the rules which used for language classification have different weights. We aim to study more about the factors that impact on language recognition. This means we need to read more linguistic literature and introduce the factors mentioned in our methods.

We conclude that it is feasible to map languages from ALID to TLID. However, there are still many questions other than dimension questions. For example, can other algorithm-information theories calculate the language distance? How does their performance compare to the zipping and the n-gram model? Also, are Sammon mapping and Shepard's interpolation the optimal options for mapping? Are there any alternative methods that perform better than Sammon mapping and Shepard's interpolation? Since there are many linguistic language classifications, is it possible to find a background truth tree that can be used for unknown text, audio and video language classification?

For VLID, it is necessary to use more dataset that can be used for training and testing Mandarin and Arabic. We also need more languages to build a bigger language distance matrix that can be used for mapping to audio and text language dimensions.

Appendix A

List of text language datasets

Table A.1: List of text languages datasets.

ISO 639-2	**Language**	**ISO 639-2**	**Language**
abk	Abkhaz	atj	Achehnese
jiv	Achuar	acu	Achuar-Shiwiar
ajg	Adja	ady	Adyghe
gax	Afaan	afk	Afrikaans
agr	Aguaruna	ccc	A'ingae
tws	Akuapem	aln	Albanian
alt	Altay	amc	Amahuaca
amr	Amarakaeri	amh	Amharic
ame	Amuesha-Yanesha	njo	Ao
arl	Arabela	arz	Arabic
arm	Armenian	ass	Asante
cni	Asháninca	cpu	Ashéninca
asm	Assamese	aii	Assyrian
awa	Awadhi	kbd	Kabardian
kwi	Awapit	aym	Aymara
aub	Bable	inz	Bahasa

azb	Azeri/Azerbaijani	bos	Bosnian
mli	Bahasa	bca	Bai
bvi	Balanda	bzc	Balinese
bgp	Balochi	bra	Bambara
bci	Baoulé/Baule	bfa	Bari
bsq	Basque	bba	Baatonum
ruw	Belorus	bem	Bemba
bng	Bengali	btb	Béti
bhj	Bhojpuri	bcy	Bichelamar
bkl	Bikol/Bicolano	boa	Bora
brt	Breton	bpr	Bugisnese
blg	Bulgarian	bms	Burmese/Myanmar
cak	Kaqohiquel	cpp	Campa
cbu	Candoshi-Shapra	cot	Caquinte
cbr	Cashibo-Cacataibo	cbs	Cashinahua
cln	Catalan	ceb	Cebuano
cbi	Chaa'pala	cjd	Chamorro
tso	Changane	cbt	Chayahuita
nyj	Chechewa	hne	Chhattisgarhi
cic	Chickasaw	fal	Fali
hak	Hakka Chinese	hlt	Matu chin
tid	Chin	csa	Chinanteco
chj	Chinanteco	chn	Chinese
cax	Chiquitano	tru	Surayt Taroyo
cjk	Chokwe	coi	Corsican
kea	Crioulo	gbc	Crioulo
hrv	Croatian	wls	Cymraeg
czc	Czech	dga	Dagaare

dag	Dagbani	gac	Dangme
dns	Danish	prs	Dari
den	Dendi	ger	Deutsch
div	Dhivehi	nav	Dine
dinka	Dinka	dyo	Diola
dyu	Dioula	tbz	Ditammari
dut	Dutch	dzo	Dzongkha/Bhutanese
edo	Edo	ibb	Efik
grk	Ellinika' (Greek)	eng	English
spn	Español (Spanish)	epo	Esperanto
est	Estonian	bsq	Euskara
eve	Even	evn	Evenki
ewe	Ewe/Eve	tws	Fante
fae	Faroese	prs	Farsi/Persian
fp	Fijian	tgl	Filipino
fin	Finnish	kng	koongo
foa	Fon	cri	Forro
frn	French	fri	Frisian
frl	Friulian	gac	Ga
gli	Gaeilge	gag	Gagauz
gls	Gàidhlig (Swedish Gaelic)	gln	Galician
gbm	Garhwali	cab	Garifuna
geo	Georgian	ger	German
gno	Gondi	dum	Gonja
grk	Greek	esg	Greenlandic
gun	Guarani	gua	Sliki
hna	Mina Cameroon	gjr	Gujarati
hat	Haitian	nyj	Nyanja/Chinyanja

hni	Hani	kkn	Hankuko
gej	Gen	hwi	Hawaiian
hbr	Hebrew	hil	Hiligaynon
hnd	Hindi	hea	Hmong
hms	Hmong	blu	Hmong-Mien
Hoc	Ho	ccx	Zhuang
hus	Huastec	hva	Huasteco
huu	Murui Huitoto	hun	Hungarian
ibb	Ibibio	ice	Icelandic
ido	Ido	ig	Igbo
ilo	Iloko/Ilocano	ind	Indonesian
ina	Interlingua	esg	Ageri Gondi
esb	Inuktitut	gle	Irish
itn	Italian	heb	Ivrit
jpn	Japanese	jav	Javanese
maz	Central Mazahua	dyo	Jola-Fogny
kbd	Kabardian	kbp	Kabyè
bjj	Kanauji	kan	Kannada
kph	Kanuri	kqn	Kaonde
pmp	Kapampangan	krl	Karelian
pwo	Karen	kar	Karen
xsm	Kasem	kas	Kashmiri
kaz	Kazakh	kjh	Khakas
khk	Khalkha	khr	Kharia
kha	Khasi	khm	Khmer
buc	Kibushi	quc	K'iche'
qug	Kichwa	kon	Kikongo
mlo	Kimbundu	nyz	Kinyamwezi

kin	Kinyarwanda	suz	Koits-Sunuwar
koi	Komi-Permian	kor	Korean
kou	Koulango	gkp	Kpelewo
hat	Kreyol	kri	Krio
Kur	Kurdish	kur	Kurmanji
kfa	Kodava	kir	Kyrgyz
lad	Ladin	lms	Lamnso'
lao	Nomlaki		
lat	Latin	lav	Latvian
lij	Ligurian	lia	Limba
lin	Lingala	lit	Lithuanian
lob	Lobiri	nds	Low German
loz	Lozi	lua	Luba-Kasai
lug	Luganda/Ganda	lun	Lunda/Chokwe-lunda
lue	Luvale	ltz	Luxembourgish
mkd	Macedonian	mad	Madurese
mag	Magahi	hun	Magyar
mai	Maithili	kde	Makonde
vmw	Makua	mlg	Malagasy
msa	Malay	mjs	Malayalam
div	Maldivian	mls	Maltese
mam	Mam	mni	Maninka
mni	Manipuri	mbf	Maori
mri	Cook Islands	aru	Mapudungun
mar	Marathi	mzm	Marshallese
mum	Marwari	mcf	Matsés
yua	Mayan	maz	Mazahua
maa	Mazateco		

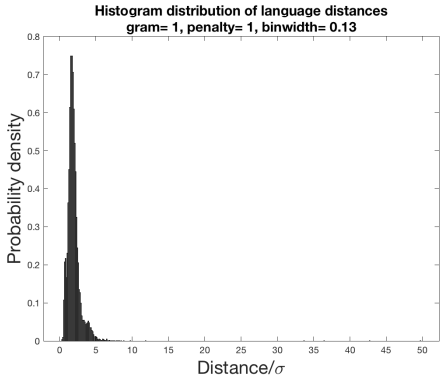
mew	Mende	mic	Mikmaq/Micmac
min	Minangkabau	miq	Miskito
mxv	Mixteco	lus	Mizo
khk	Mongolian	mhm	Mooré/More
moz	Mozarabic	unr	Mundari
oto	Ñahñú	nhn	Nahuatl
gld	Nanai	nav	Navajo
nel	Ndebele	dut	Nederlands
yrk	Nenets	nep	Nepali
nio	Nganasan	nba	Ngangela
pcm	Nigerian	Nivkh	Nivkh
not	Nomatsiguenga	srt	Northern
nrr	Norwegian	nrn	Norwegian
nus	Nuer	nyz	Nyamwezi
nze	Nzema	oki	Ogiek
ojb	Ojibway	ory	Oriya
gax	Oromiffa	ose	Osetin
kua	Oshiwambo		
lot	Otuho	pbb	Paez
sey	Pai Koka	plu	Palauan
pap	Papiamentu	pbu	Pashto/Pakhto
tsz	Purhépecha	fum	Peuhl
frn	Picard	pis	Pijin
ppl	Pipil	pql	Polish
pnf	Ponapean	por	Portuguese
pro	Prouvençau	fum	Pulaar
fuf	Pular	pnj	Punjabi/Panjabi
qec	Quechua	kek	Q'echi/Kekchi

raj	Rajasthani	rrt	Rarotongan
rhe	Rhaeto-Romance	rmn	Romani
rum	Romanian	koo	Konjo
rud	Rundi/Kirundi	nyn	Runyankore-rukiga/Nkore-kiga
rus	Russian	lpi	Sami/Lappish
eml	Sammarinese	smy	Samoan
saj	Sango	skt	Sanskrit
sat	Santhali	zro	Sapara atupama
skr	Saraiki	srd	Sardinian
hns	Carbbean Hindustani	sco	Scots
gls	Scottish Gaelic	ses	Seereer
srp	Serbian	jiv	Shuar
crs	Seselwa	sjn	Shan
mcd	Sharanahua	shk	Shilluk
swb	Shimaore	shp	Shipibo-Conibo
shd	Shona	cjs	Shor
sja	Sia Pedee	snd	Sindhi
snh	Sinhala	swz1	Siswati
slo	Slovak	slv	Slovenian
som	Somali	snn	Soninké
sso	Southern Sotho	spn	Spanish
sua	Sukuma	suo	Sundanese
fin	Suomi	sus	Sussu
swa	Swahili/Kiswahili	crm	Swampy
swd	Swedish	tht	Tahitian
pet	Tajik	tly	Talysh
taj	Tamang	taq	Tamasheq
tzm	Tamazight	tcv	Tamil

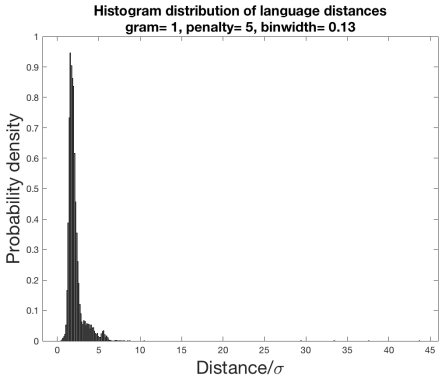
ttr	Tatar	cjk	Tchocwe
ttm	Tetum	thj	Thai
tej	Temne	tic	Tibetan
tca	Ticuna	tgn	Tigrinya
tiv	Tiv	tob	Toba
toj	Tojol-a'b'al	pdg	Tok Pisin
top	Totonaco	tru	Trukese
cof	Tsafiki	lub	Tshiluba
tsh	Venda	trk	Turkish
tck	Turkmen	tyv	Tuvan
tzc1	Tzeltal	tzc	Tzotzil
uig	Uighur	oaa	Uilta
ukr	Ukrainian	mnf	Umbundu
ura	Urarina	urd	Urdu
uzb1	Uzbek	frn	Walloon/Wallon
vai	Vai	vec	Venetian
vep	Veps	vie	Vietnamese
ako	Wama	auc	Wao
wry	Waray	guc	Wayuu
wls	Welsh	tsw	Western Sotho
wol	Wolof	xos	Xhosa
yad	Yagua	sah	Yakut
guu	Yanomamö	yao	Yao
yps	Yapese	iii	Yi
ydd	Yiddish	yor	Yoruba
ykg	Yukagir	zuu	Zulu
zap	Zapoteco		

Appendix B

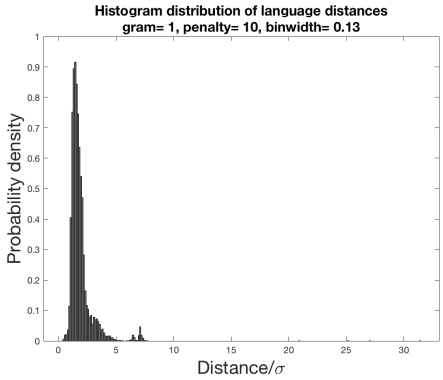
Histogram diagrams for text n-gram



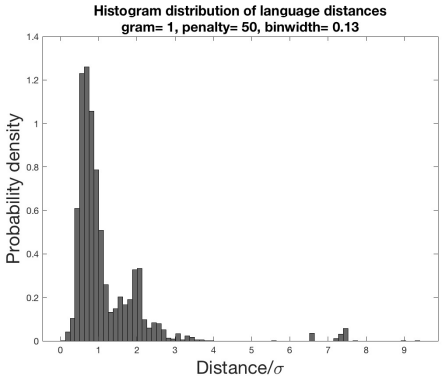
(a) Histogram distribution for penalty = 1



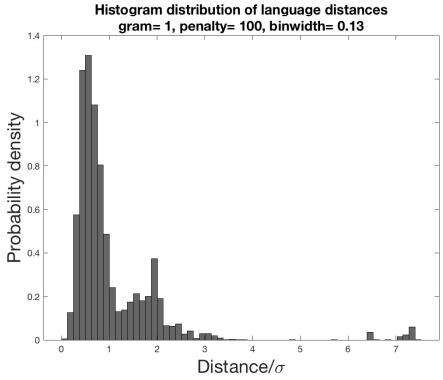
(b) Histogram distribution for penalty = 5



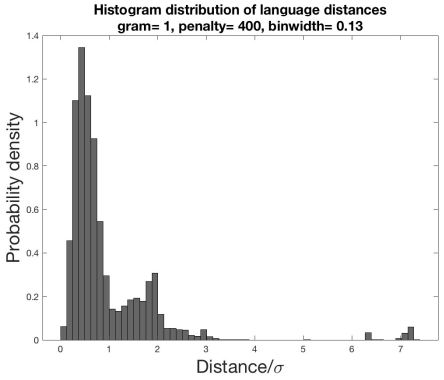
(c) Histogram distribution for penalty = 10



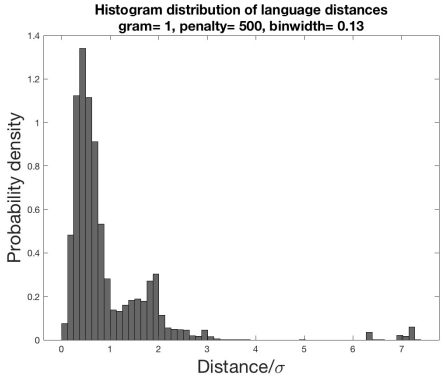
(d) Histogram distribution for penalty = 50



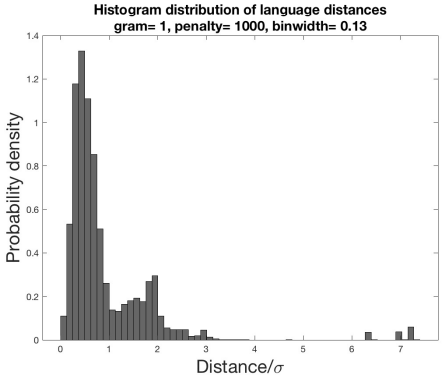
(e) Histogram distribution for penalty = 100



(f) Histogram distribution for penalty = 400

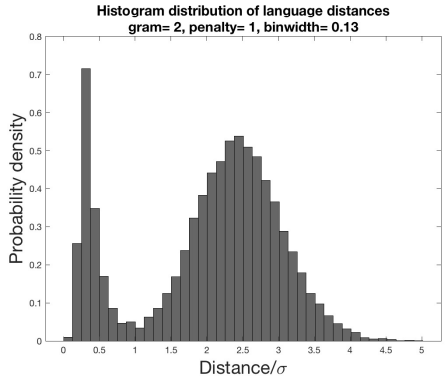


(g) Histogram distribution for penalty = 500

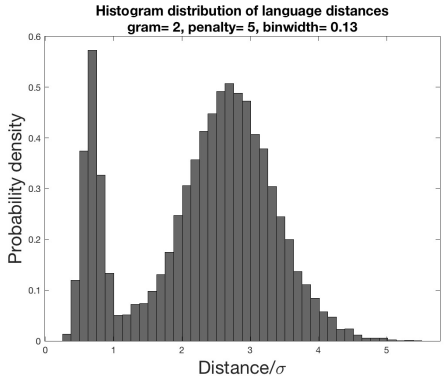


(h) Histogram distribution for penalty = 1000

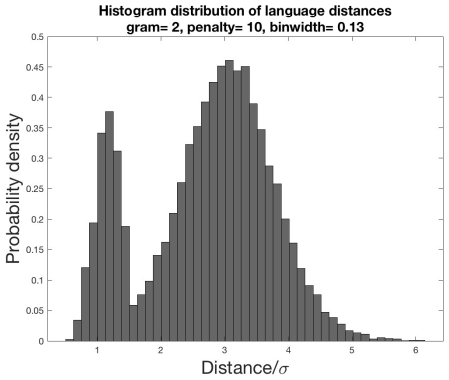
Figure B.1: Histogram distribution for 1-grams. The x-axis is the distance D/σ. The y-axis is the probability density. The binsize is the $w/\sigma = 0.13$.



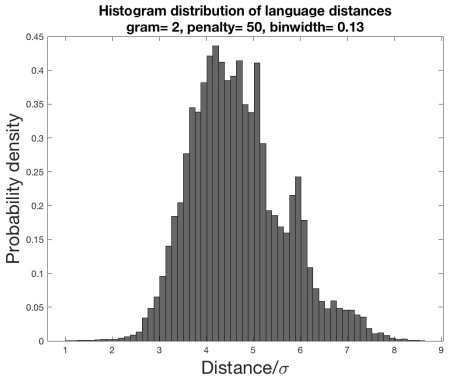
(a) Histogram distribution for penalty = 1



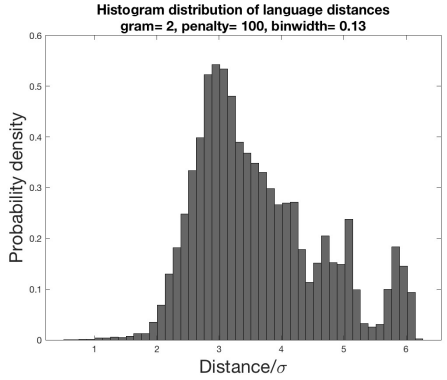
(b) Histogram distribution for penalty = 5



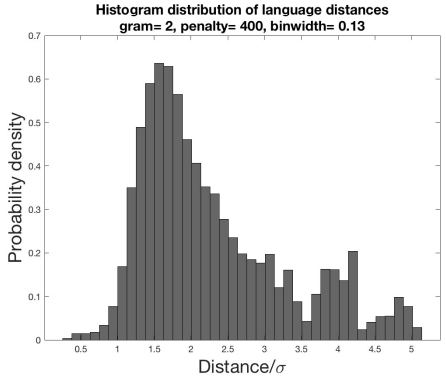
(c) Histogram distribution for penalty = 10



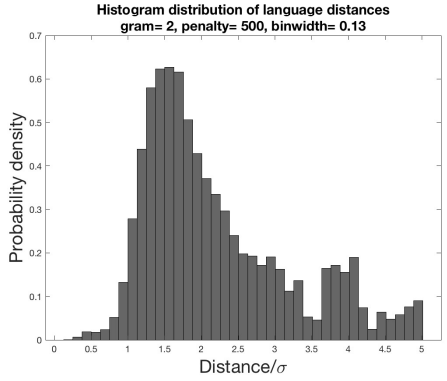
(d) Histogram distribution for penalty = 50



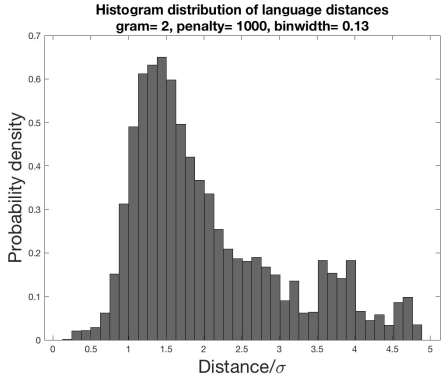
(e) Histogram distribution for penalty = 100



(f) Histogram distribution for penalty = 400

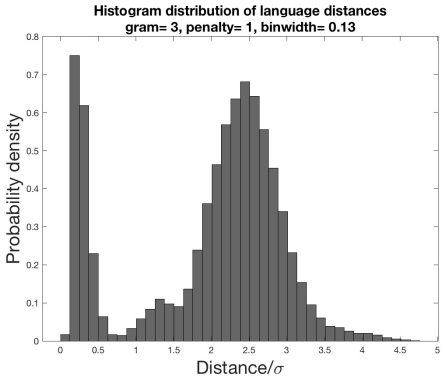


(g) Histogram distribution for penalty = 500

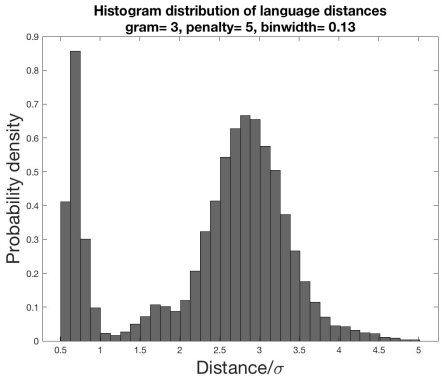


(h) Histogram distribution for penalty = 1000

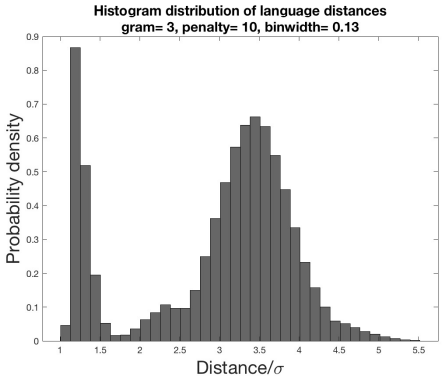
Figure B.2: Histogram distribution for 2-grams. The x-axis is the distance D/σ. The y-axis is the probability density. The binsize is the $w/\sigma = 0.13$.



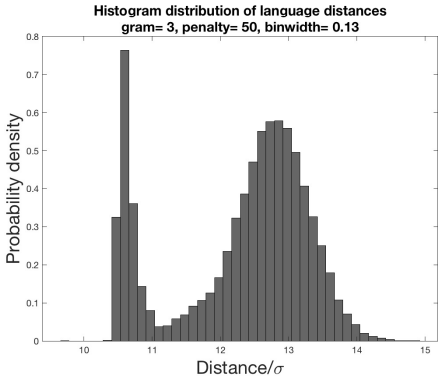
(a) Histogram distribution for penalty = 1



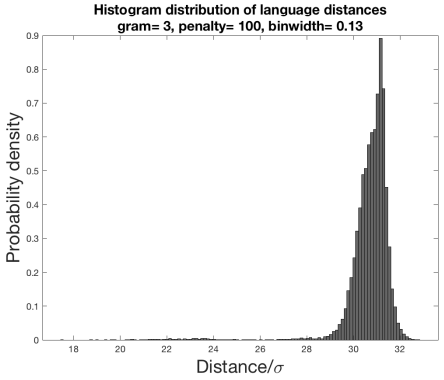
(b) Histogram distribution for penalty = 5



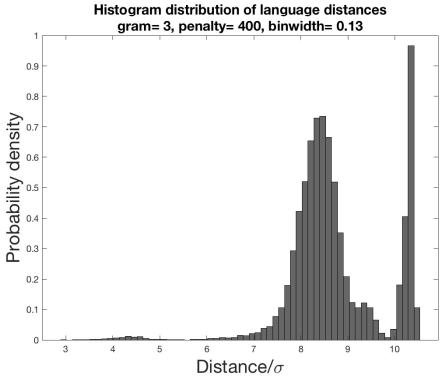
(c) Histogram distribution for penalty = 10



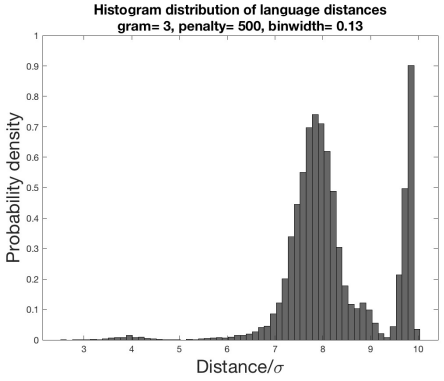
(d) Histogram distribution for penalty = 50



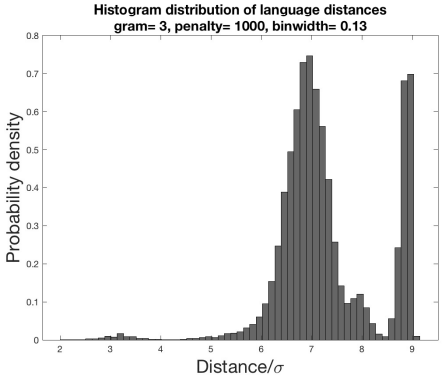
(e) Histogram distribution for penalty = 100



(f) Histogram distribution for penalty = 400

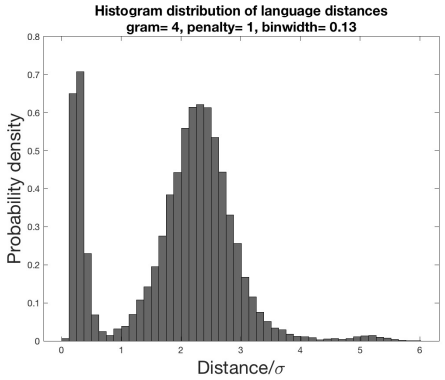


(g) Histogram distribution for penalty = 500

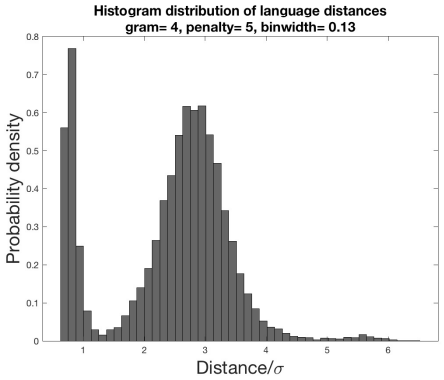


(h) Histogram distribution for penalty = 1000

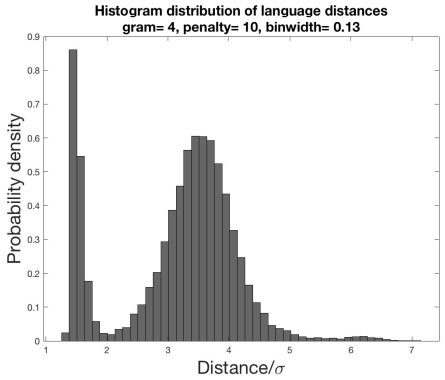
Figure B.3: Histogram distribution for 3-grams. The x-axis is the distance D/σ. The y-axis is the probability density. The binsize is the $w/\sigma = 0.13$.



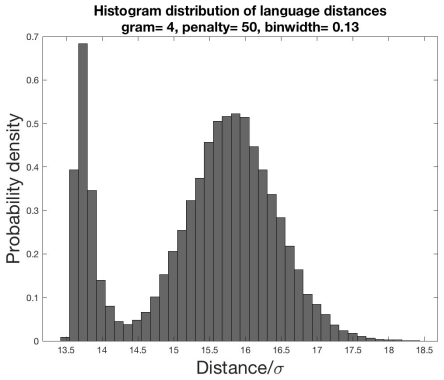
(a) Histogram distribution for penalty = 1



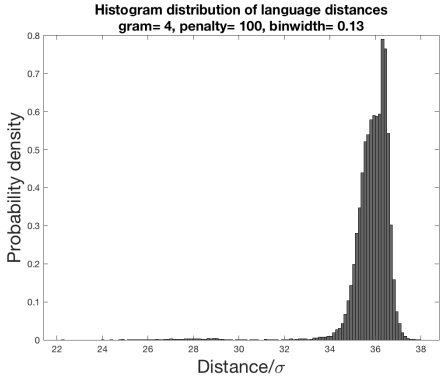
(b) Histogram distribution for penalty = 5



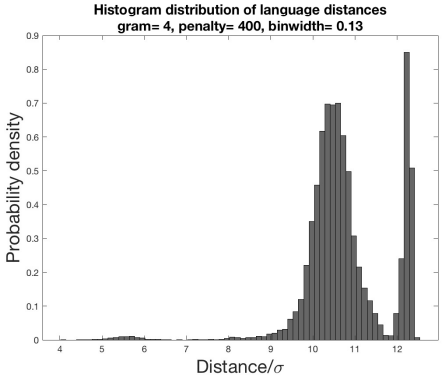
(c) Histogram distribution for penalty = 10



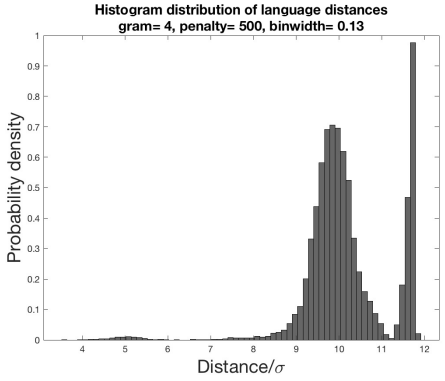
(d) Histogram distribution for penalty = 50



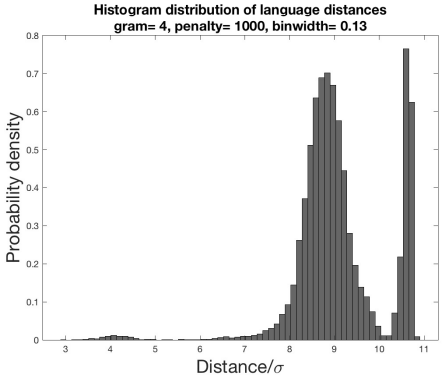
(e) Histogram distribution for penalty = 100



(f) Histogram distribution for penalty = 400

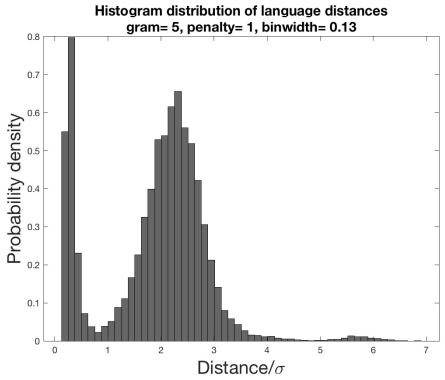


(g) Histogram distribution for penalty = 500

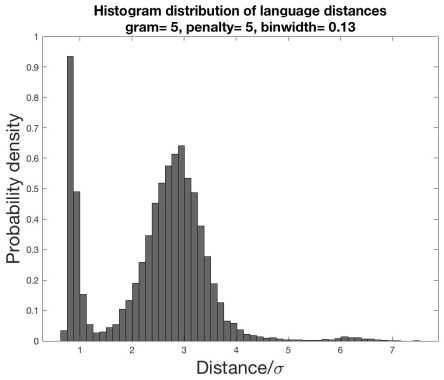


(h) Histogram distribution for penalty = 1000

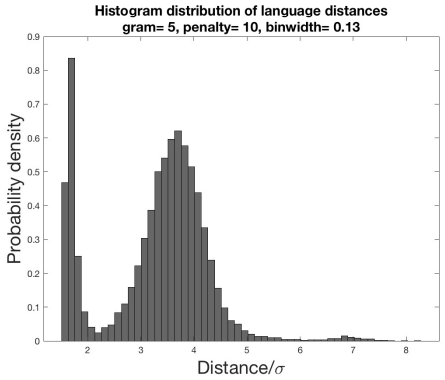
Figure B.4: Histogram distribution for 4-grams. The x-axis is the distance D/σ. The y-axis is the probability density. The binsize is the $w/\sigma = 0.13$.



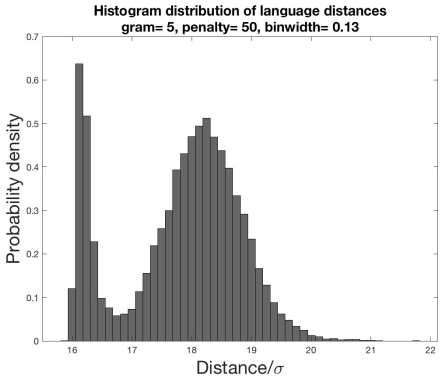
(a) Histogram distribution for penalty = 1



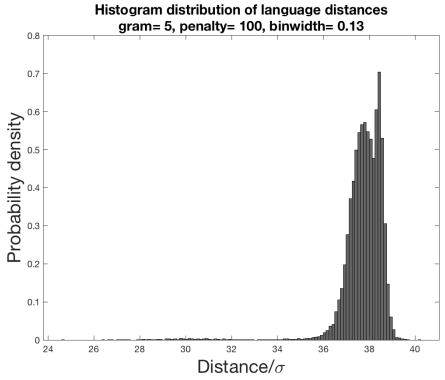
(b) Histogram distribution for penalty = 5



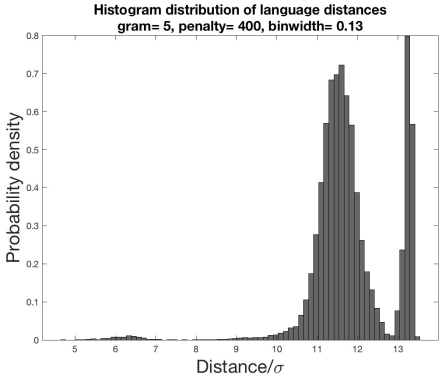
(c) Histogram distribution for penalty = 10



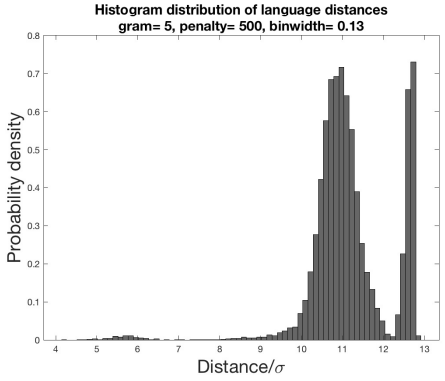
(d) Histogram distribution for penalty = 50



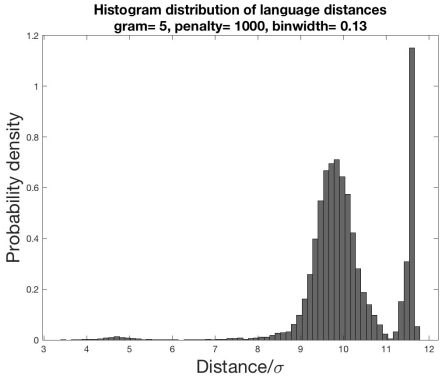
(e) Histogram distribution for penalty = 100



(f) Histogram distribution for penalty = 400



(g) Histogram distribution for penalty = 500

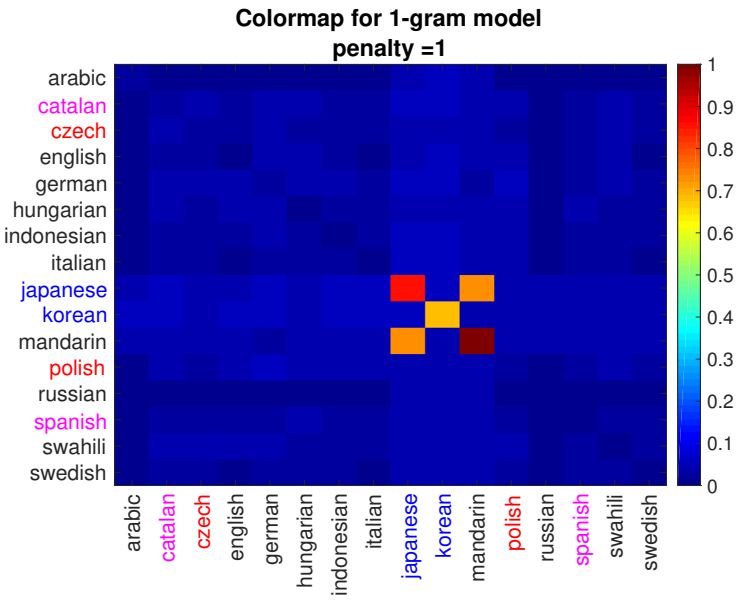


(h) Histogram distribution for penalty = 1000

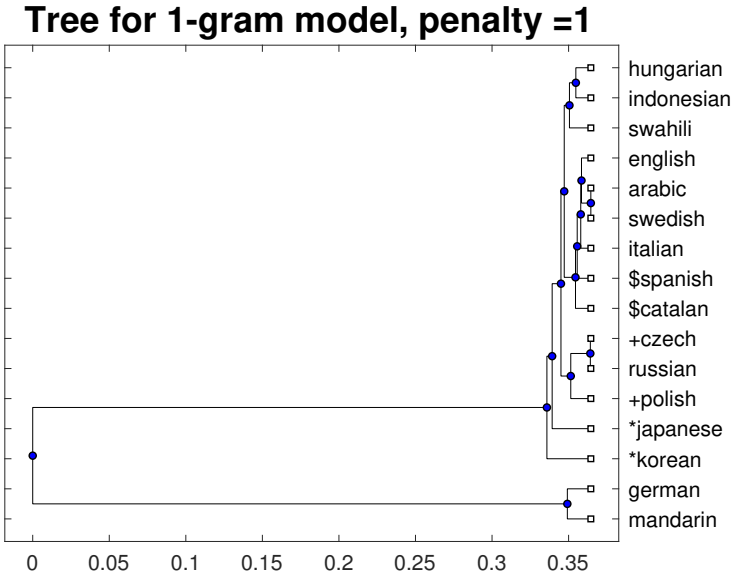
Figure B.5: Histogram distribution for 5-grams. The x-axis is the distance D/σ. The y-axis is the probability density. The binsize is the $w/\sigma = 0.13$.

Appendix C

TLID n-gram color maps and language trees

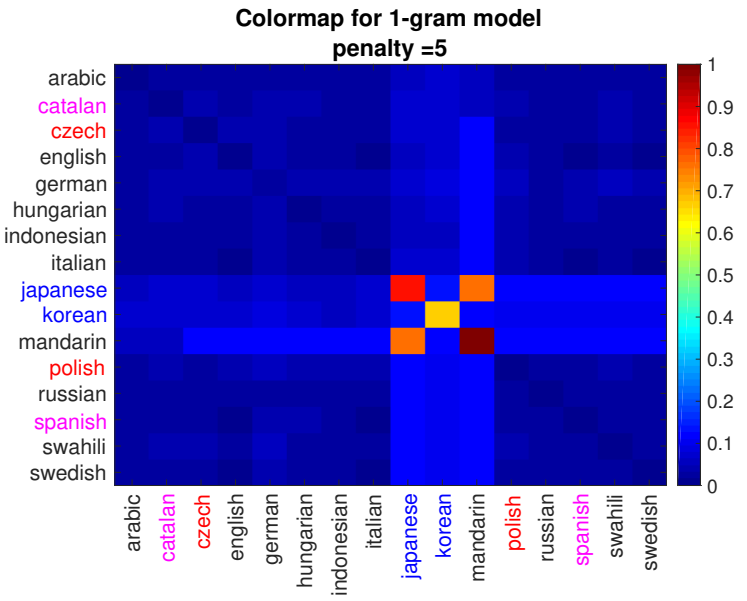


(a) Colormap of uni-gram

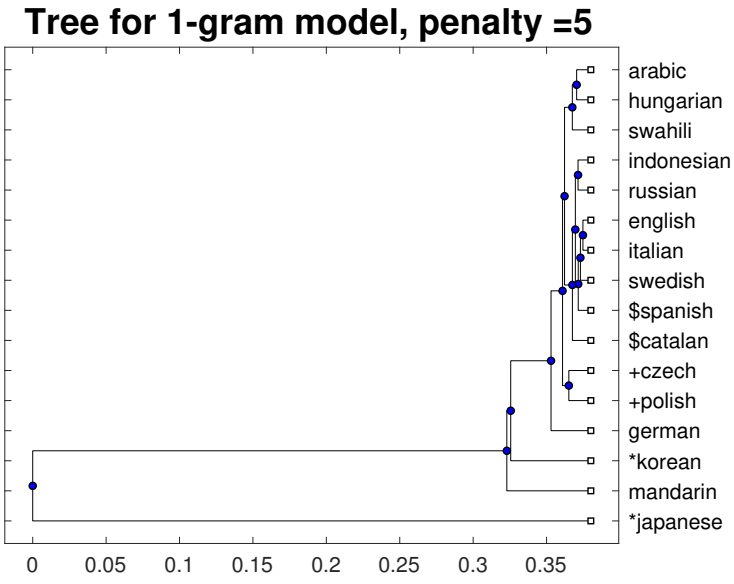


(b) Tree structure of uni-gram

Figure C.1: The 16 UNDHR text language distances results of uni-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 1. Figure C.1(a) shows the colormap of the language distance variations and Figure C.1(b) shows the language tree which is built by the distances. The colour variation in Figure C.1(a) shows the pairwise distances between languages.

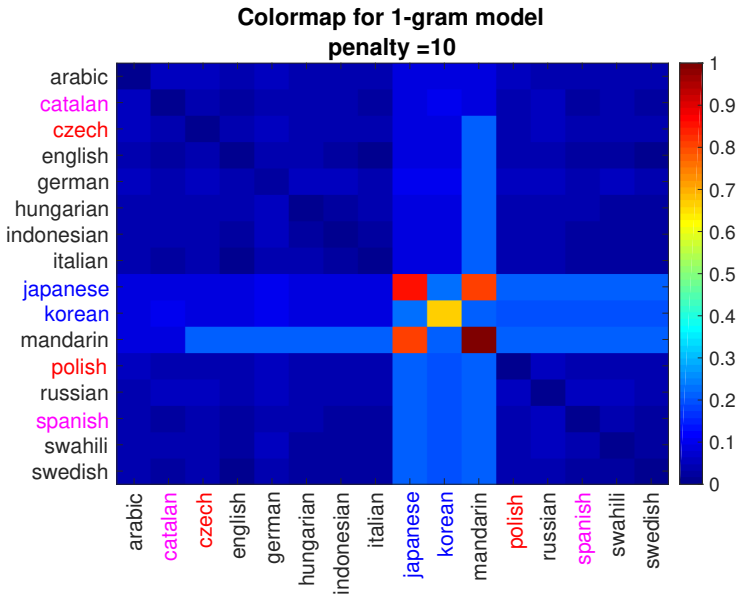


(a) Colormap of uni-gram

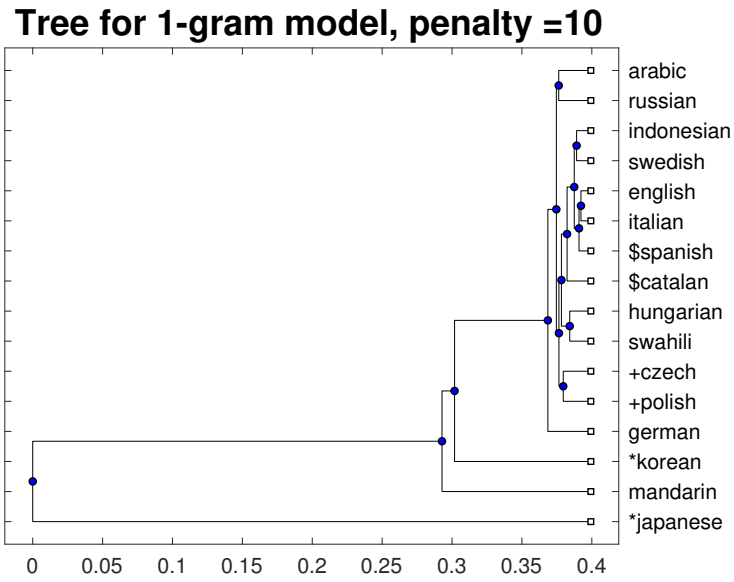


(b) Tree structure of uni-gram

Figure C.2: The 16 UNDHR text language distances results of uni-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 5. Figure C.2(a) shows the colormap of the language distance variations and Figure C.2(b) shows the language tree which is built by the distances. The colour variation in Figure C.2(a) shows the pairwise distances between languages.

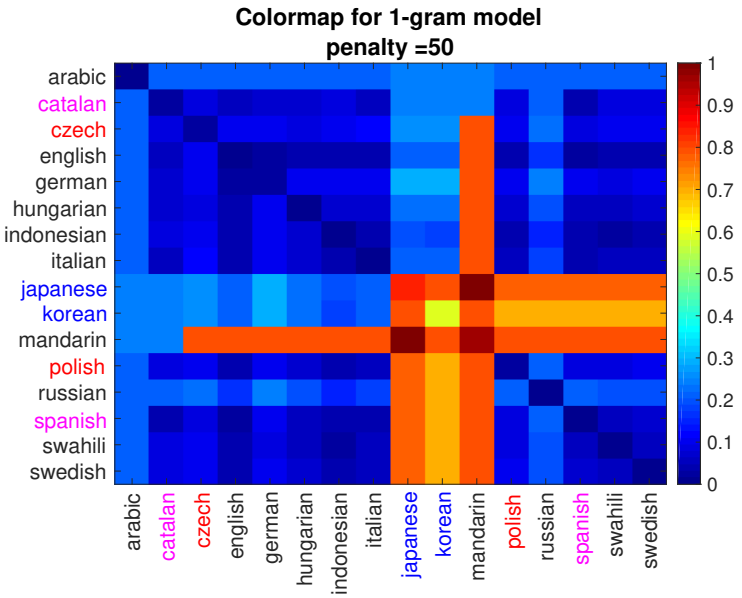


(a) Colormap of uni-gram

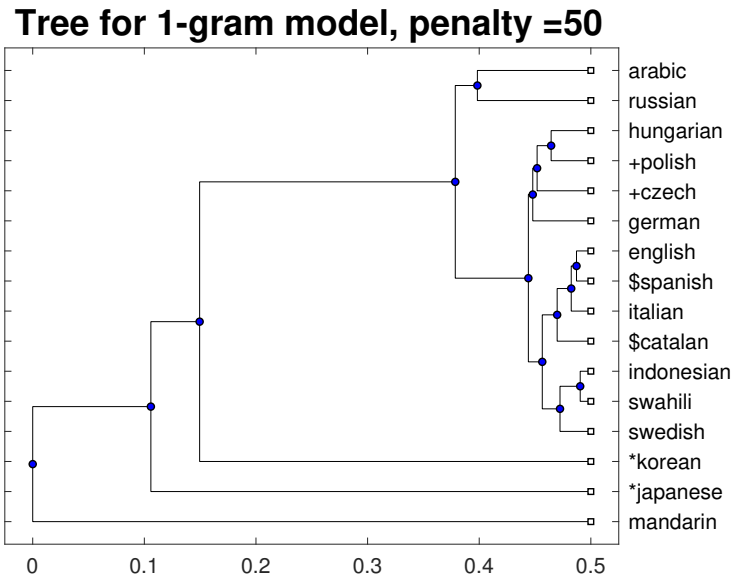


(b) Tree structure of uni-gram

Figure C.3: The 16 UNDHR text language distances results of uni-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 10. Figure C.3(a) shows the colormap of the language distance variations and Figure C.3(b) shows the language tree which is built by the distances. The colour variation in Figure C.3(a) shows the pairwise distances between languages.

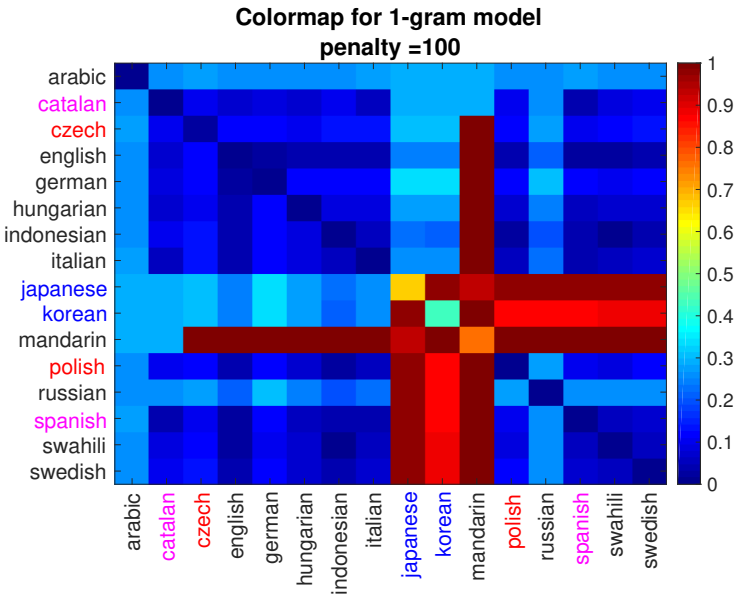


(a) Colormap of uni-gram

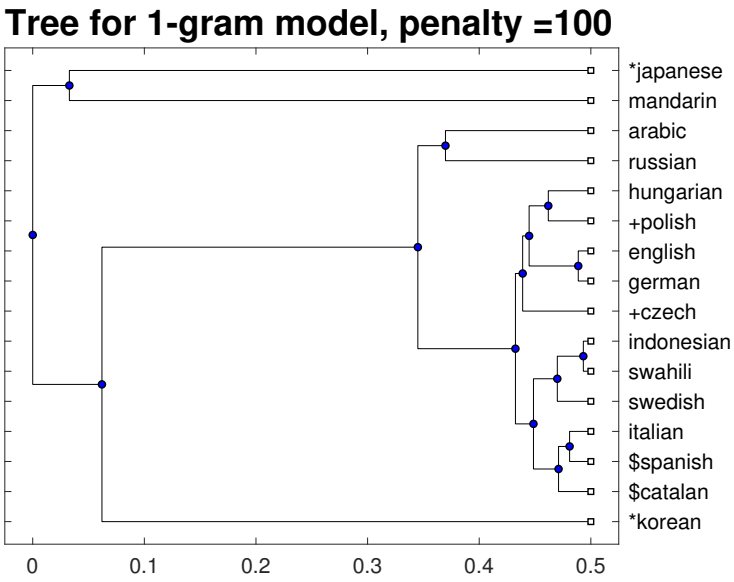


(b) Tree structure of uni-gram

Figure C.4: The 16 UNDHR text language distances results of uni-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 50. Figure C.4(a) shows the colormap of the language distance variations and Figure C.4(b) shows the language tree which is built by the distances. The colour variation in Figure C.4(a) shows the pairwise distances between languages.

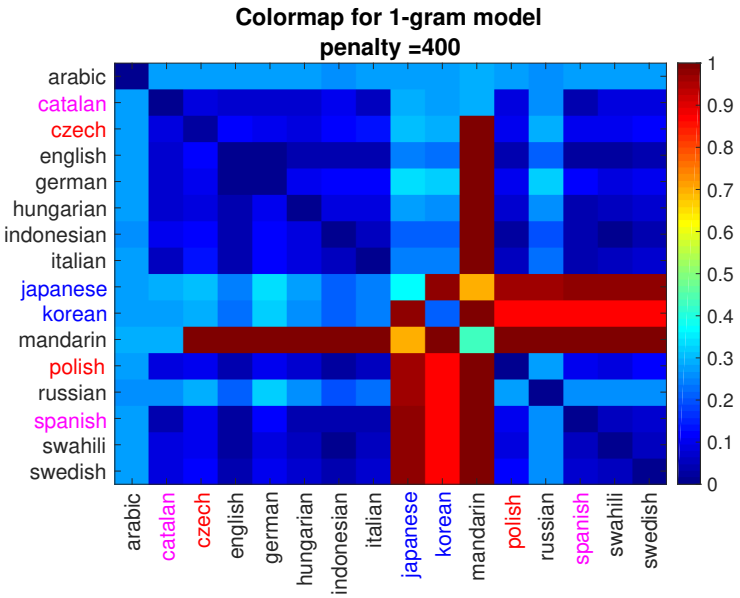


(a) Colormap of uni-gram

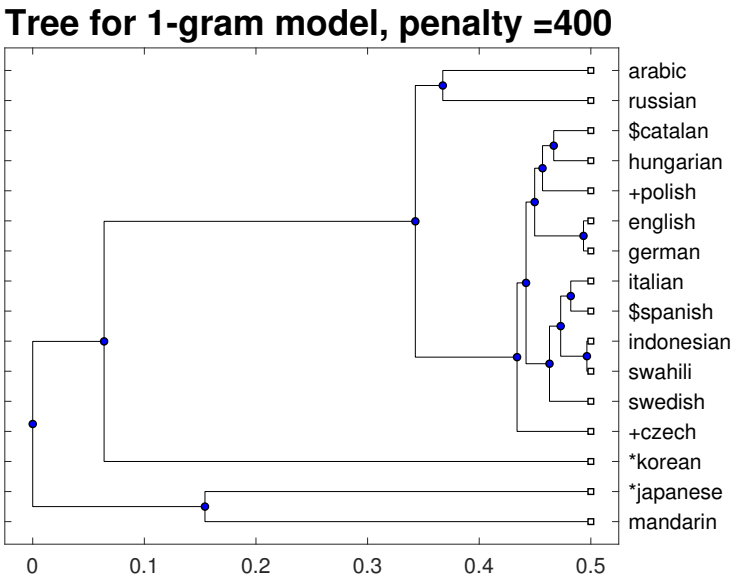


(b) Tree structure of uni-gram

Figure C.5: The 16 UNDHR text language distances results of uni-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 100. Figure C.5(a) shows the colormap of the language distance variations and Figure C.5(b) shows the language tree which is built by the distances. The colour variation in Figure C.5(a) shows the pairwise distances between languages.

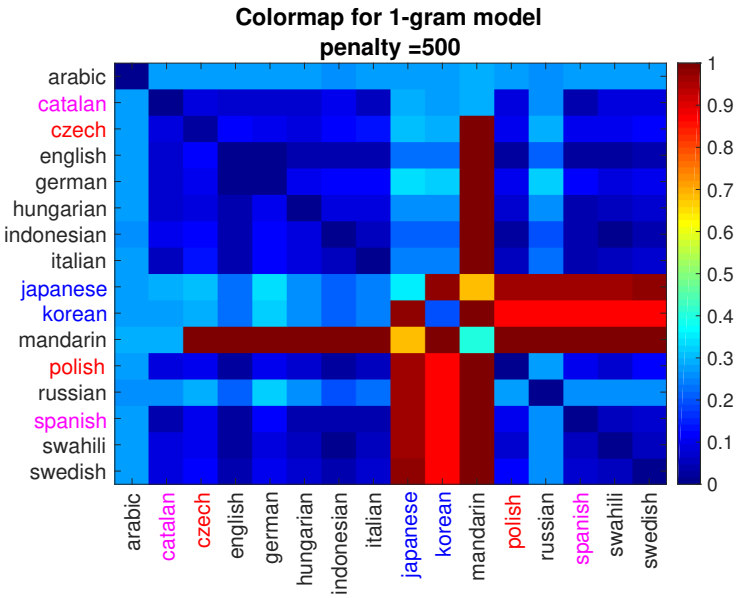


(a) Colormap of uni-gram

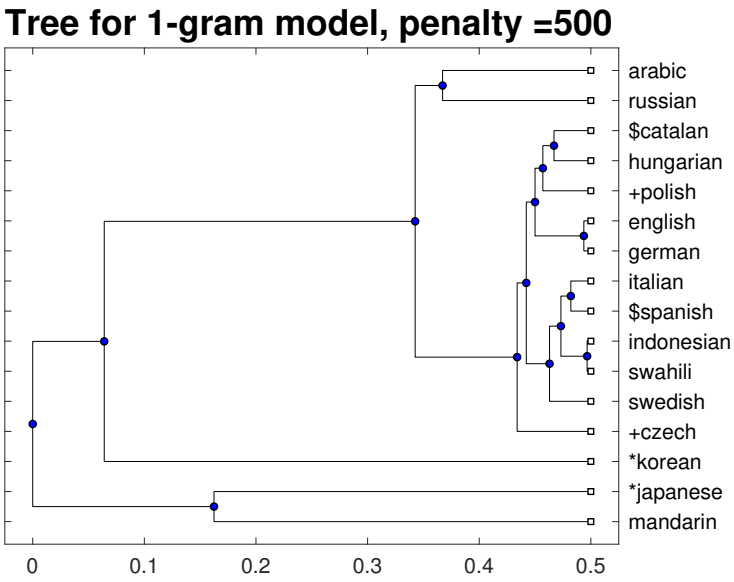


(b) Tree structure of uni-gram

Figure C.6: The 16 UNDHR text language distances results of uni-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 400. Figure C.6(a) shows the colormap of the language distance variations and Figure C.6(b) shows the language tree which is built by the distances. The colour variation in Figure C.6(a) shows the pairwise distances between languages.

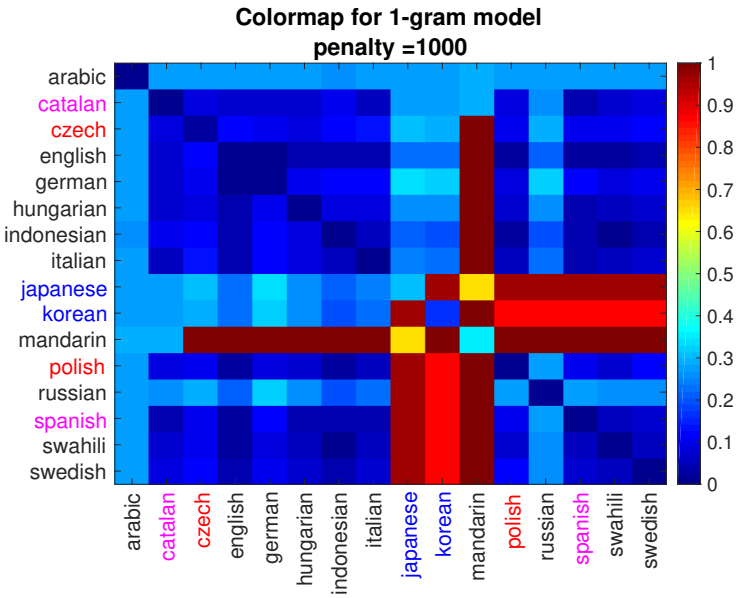


(a) Colormap of uni-gram

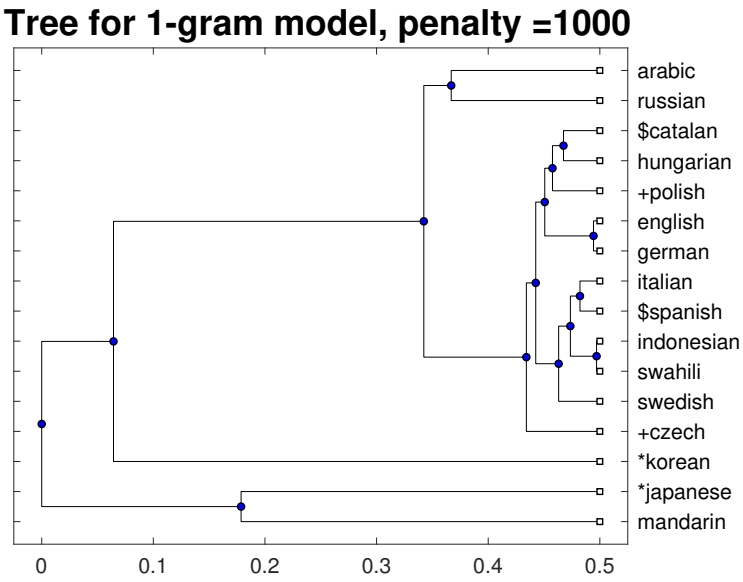


(b) Tree structure of uni-gram

Figure C.7: The 16 UNDHR text language distances results of uni-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 500. Figure C.7(a) shows the colormap of the language distance variations and Figure C.7(b) shows the language tree which is built by the distances. The colour variation in Figure C.7(a) shows the pairwise distances between languages.

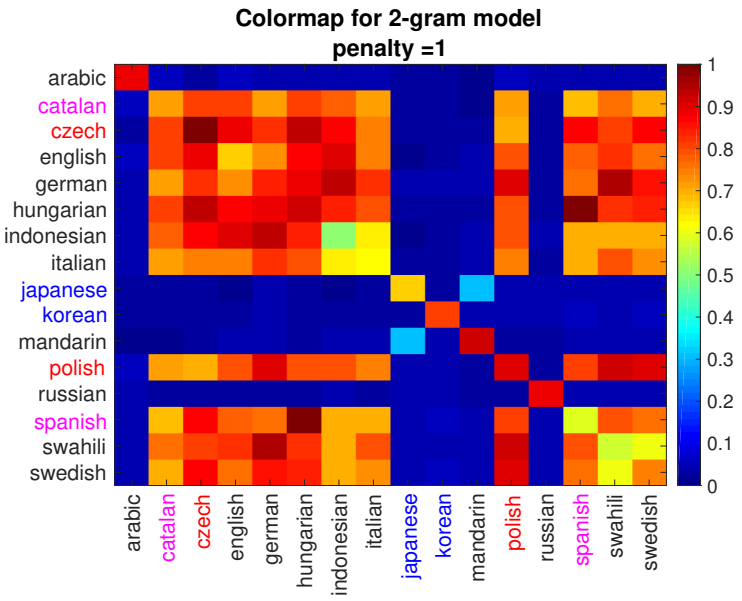


(a) Colormap of uni-gram

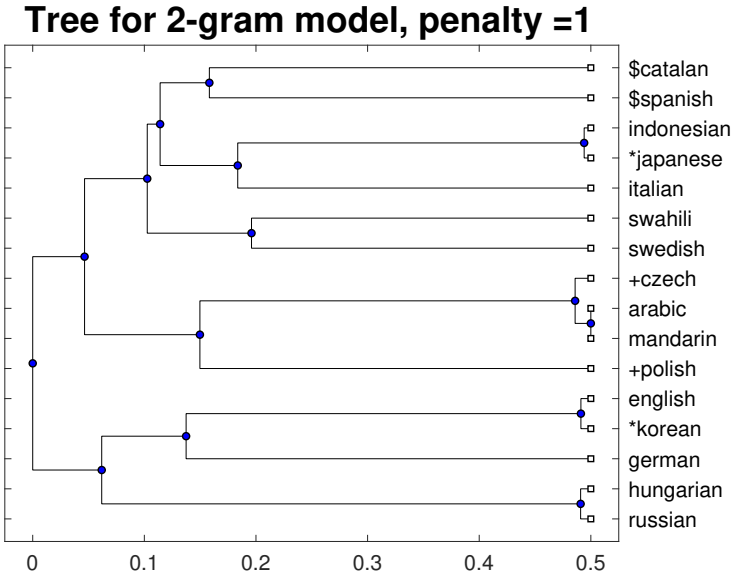


(b) Tree structure of uni-gram

Figure C.8: The 16 UNDHR text language distances results of uni-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 1000. Figure C.8(a) shows the colormap of the language distance variations and Figure C.8(b) shows the language tree which is built by the distances. The colour variation in Figure C.8(a) shows the pairwise distances between languages.

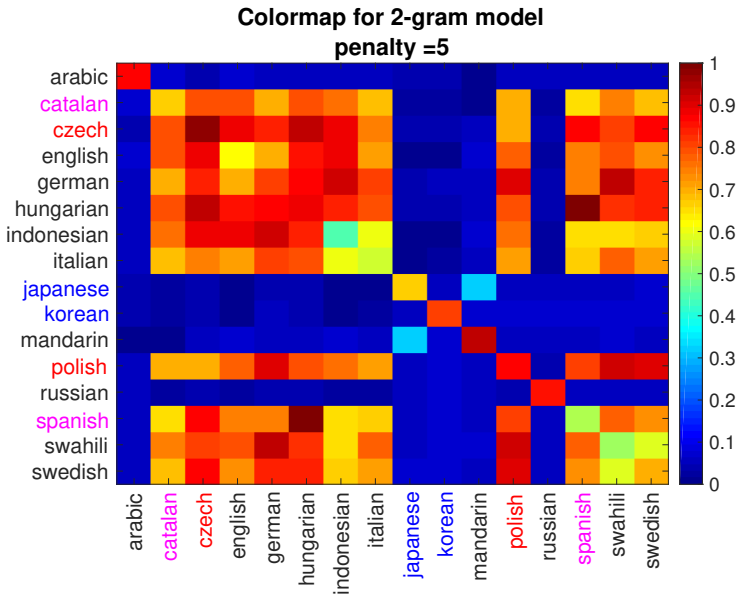


(a) Colormap of bi-gram

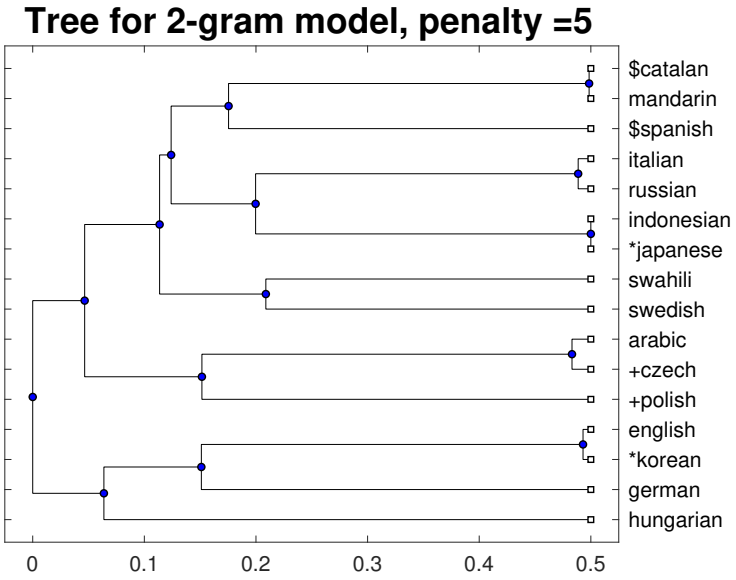


(b) Tree structure of bi-gram

Figure C.9: The 16 UNDHR text language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 1. Figure C.9(a) shows the colormap of the language distance variations and Figure C.9(b) shows the language tree which is built by the distances. The colour variation in Figure C.9(a) shows the pairwise distances between languages.

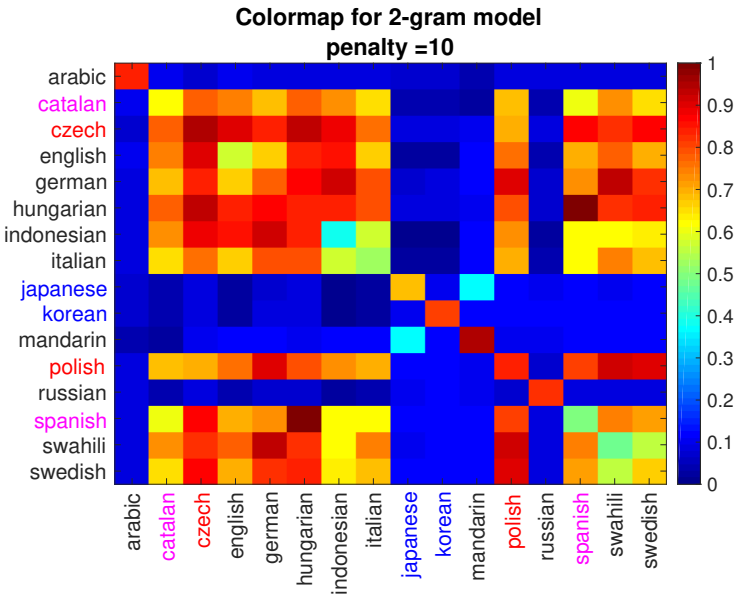


(a) Colormap of bi-gram

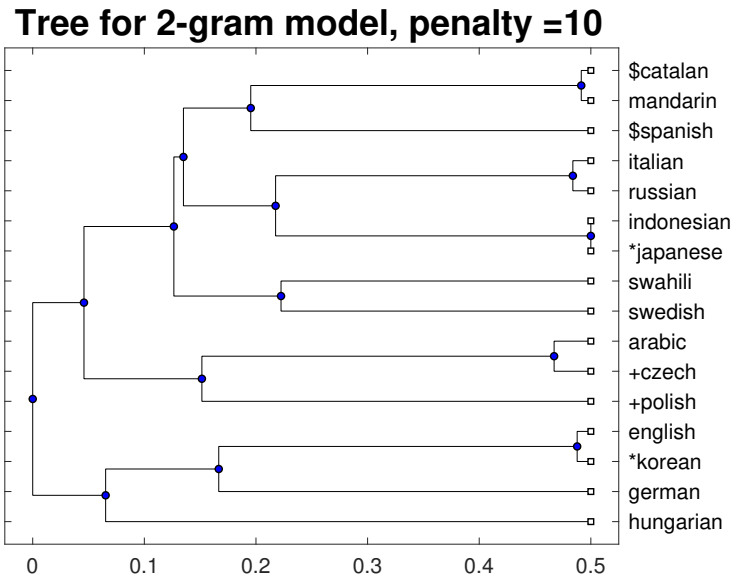


(b) Tree structure of bi-gram

Figure C.10: The 16 UNDHR text language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 5. Figure C.10(a) shows the colormap of the language distance variations and Figure C.10(b) shows the language tree which is built by the distances. The colour variation in Figure C.10(a) shows the pairwise distances between languages.

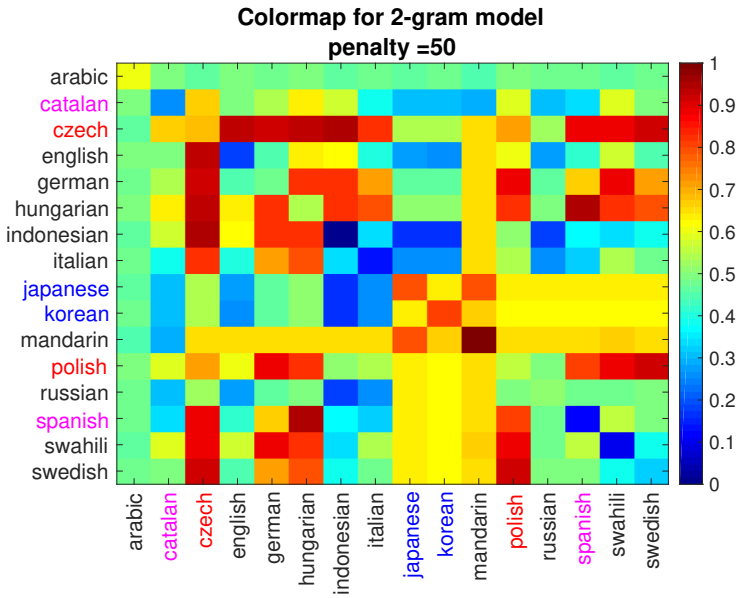


(a) Colormap of bi-gram

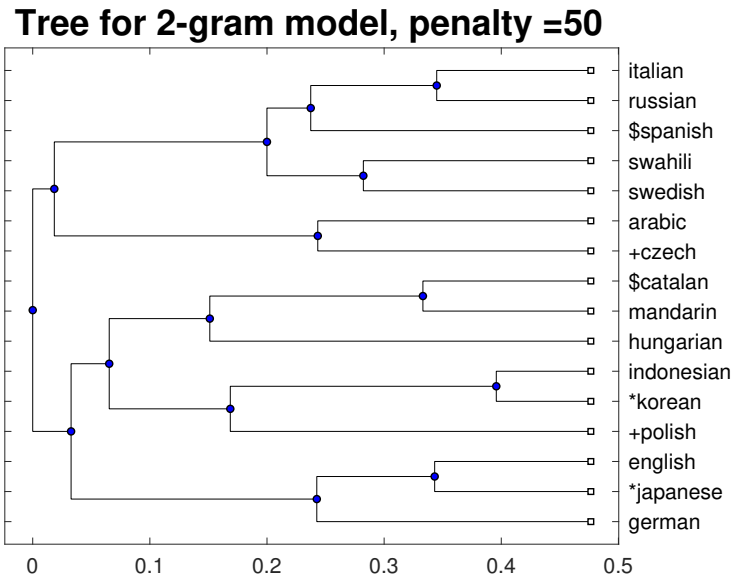


(b) Tree structure of bi-gram

Figure C.11: The 16 UNDHR text language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 10. Figure C.11(a) shows the colormap of the language distance variations and Figure C.11(b) shows the language tree which is built by the distances. The colour variation in Figure C.11(a) shows the pairwise distances between languages.

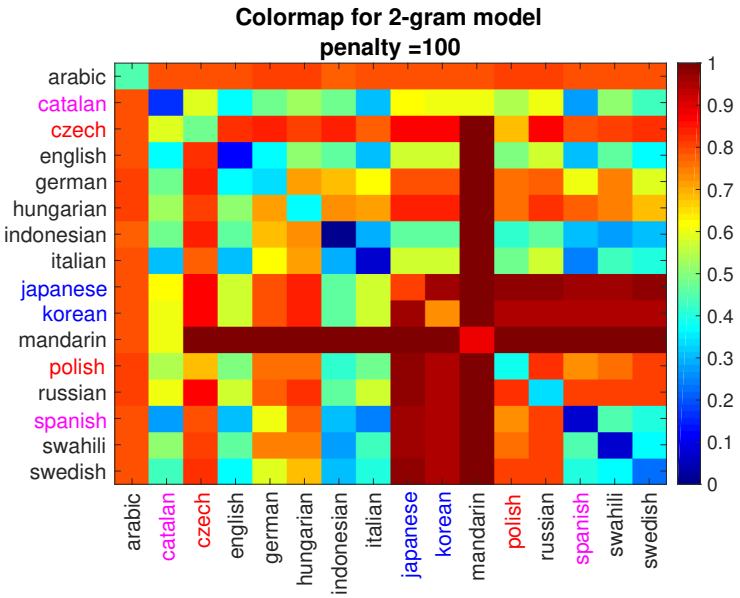


(a) Colormap of bi-gram

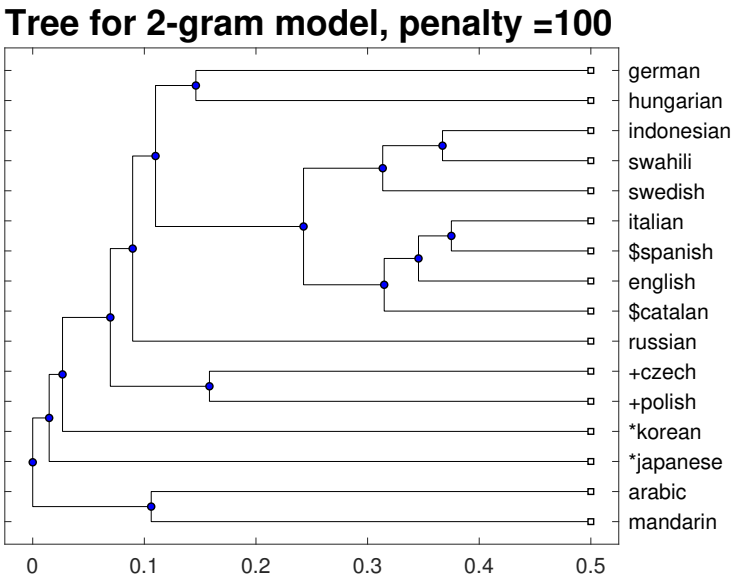


(b) Tree structure of bi-gram

Figure C.12: The 16 UNDHR text language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 50. Figure C.12(a) shows the colormap of the language distance variations and Figure C.12(b) shows the language tree which is built by the distances. The colour variation in Figure C.12(a) shows the pairwise distances between languages.

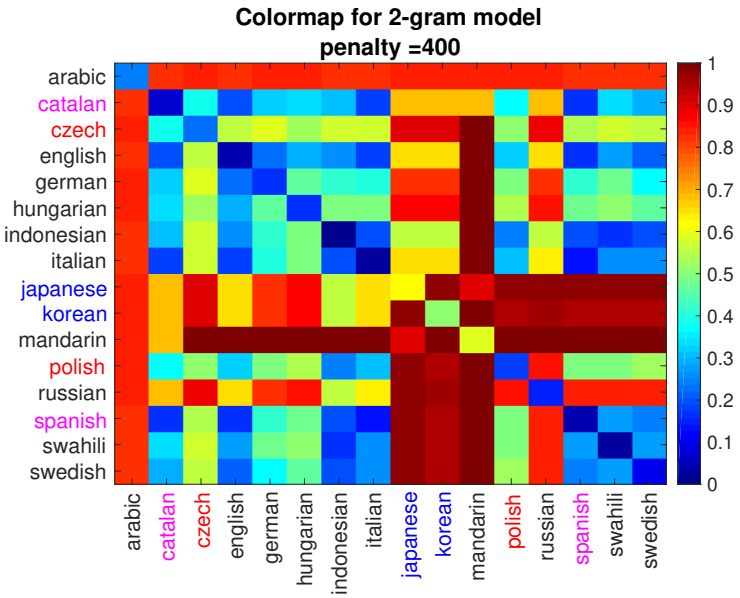


(a) Colormap of bi-gram

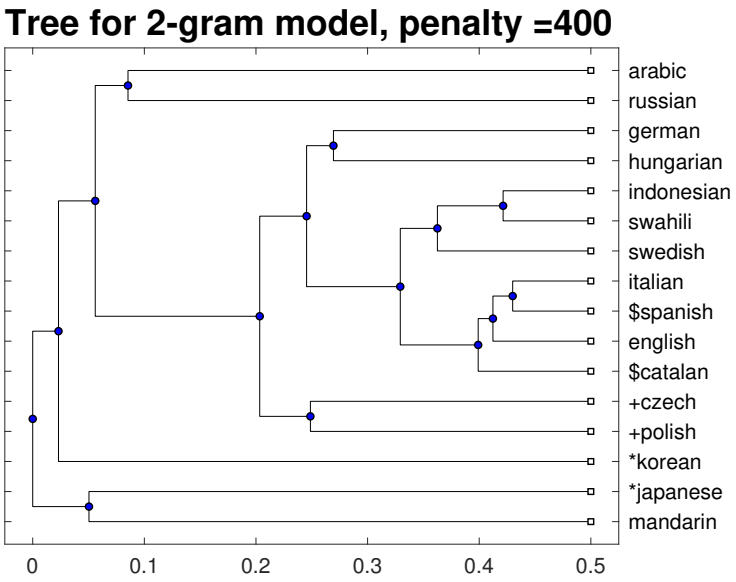


(b) Tree structure of bi-gram

Figure C.13: The 16 UNDHR text language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 100. Figure C.13(a) shows the colormap of the language distance variations and Figure C.13(b) shows the language tree which is built by the distances. The colour variation in Figure C.13(a) shows the pairwise distances between languages.

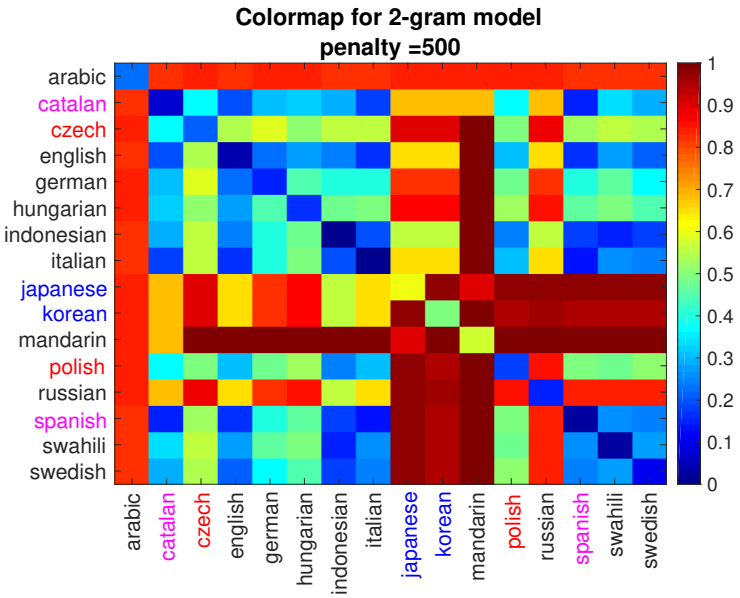


(a) Colormap of bi-gram

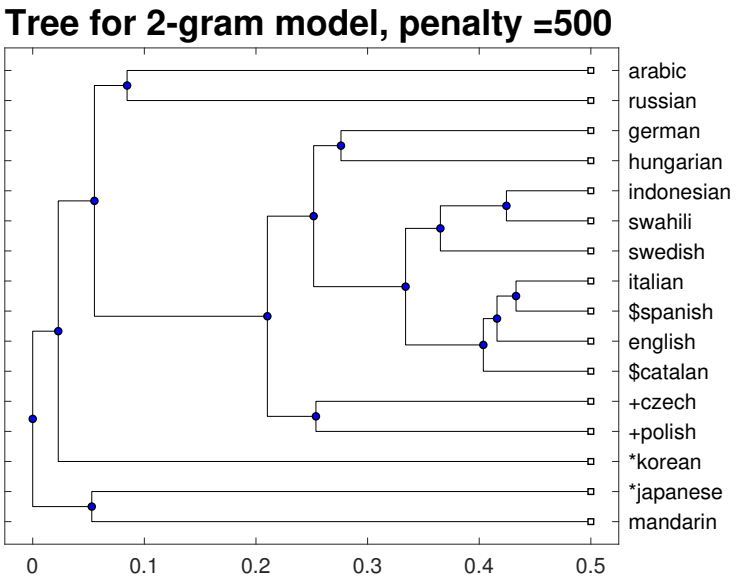


(b) Tree structure of bi-gram

Figure C.14: The 16 UNDHR text language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 400. Figure C.14(a) shows the colormap of the language distance variations and Figure C.14(b) shows the language tree which is built by the distances. The colour variation in Figure C.14(a) shows the pairwise distances between languages.

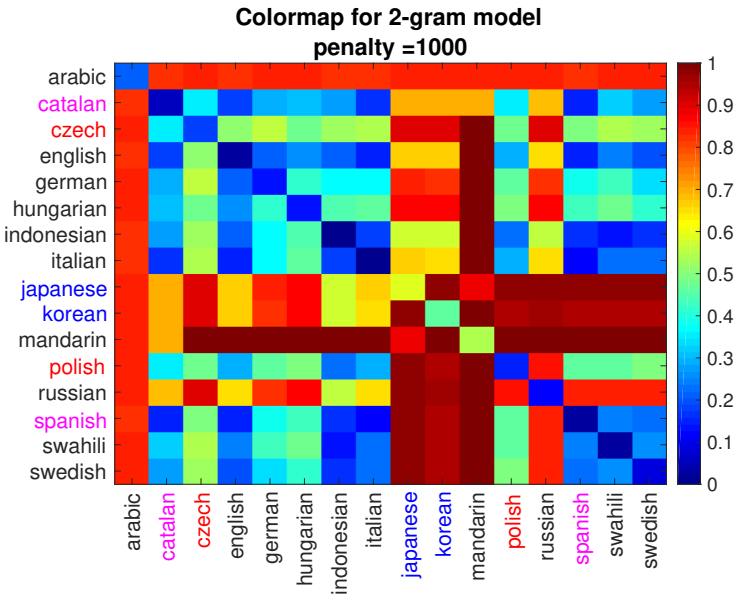


(a) Colormap of bi-gram

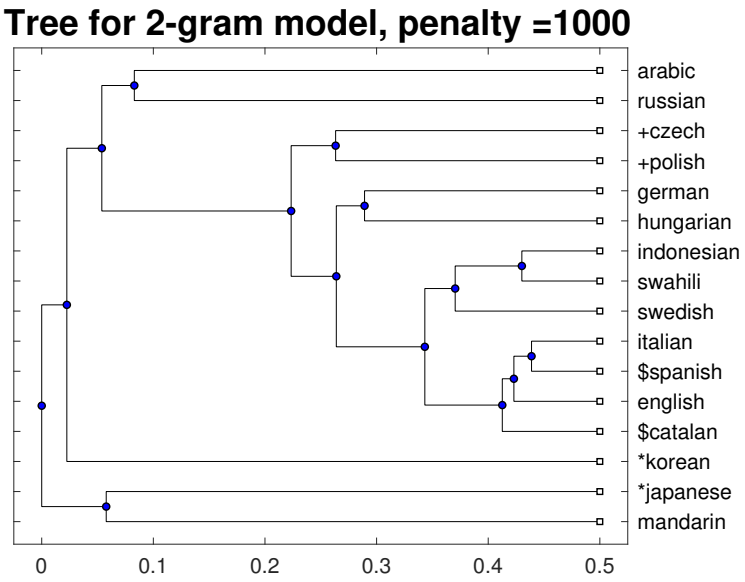


(b) Tree structure of bi-gram

Figure C.15: The 16 UNDHR text language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 500. Figure C.15(a) shows the colormap of the language distance variations and Figure C.15(b) shows the language tree which is built by the distances. The colour variation in Figure C.15(a) shows the pairwise distances between languages.

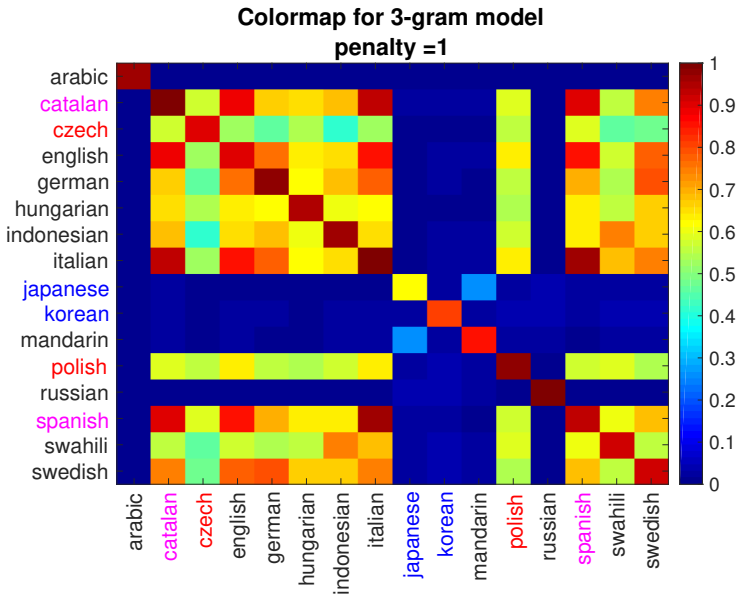


(a) Colormap of bi-gram

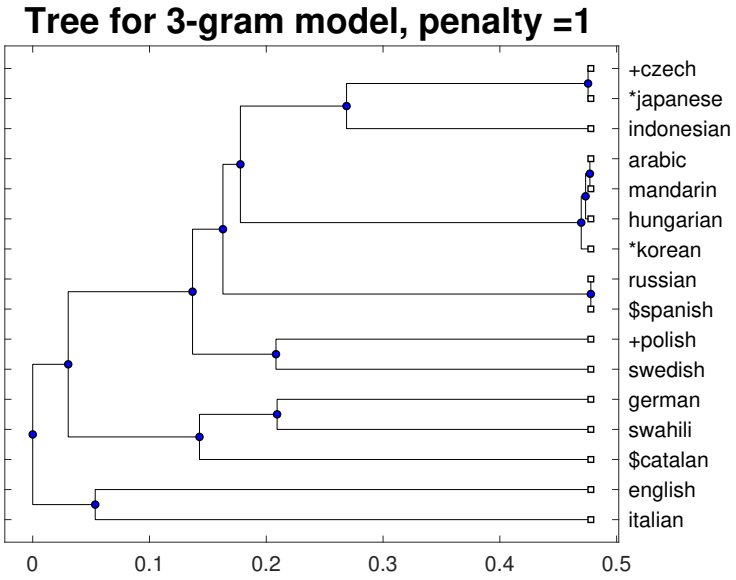


(b) Tree structure of bi-gram

Figure C.16: The 16 UNDHR text language distances results of bi-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 1000. Figure C.16(a) shows the colormap of the language distance variations and Figure C.16(b) shows the language tree which is built by the distances. The colour variation in Figure C.16(a) shows the pairwise distances between languages.

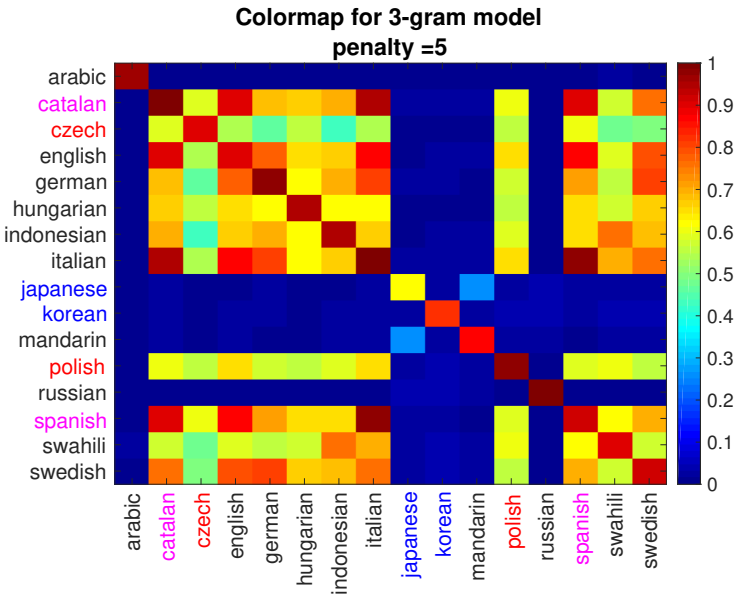


(a) Colormap of tri-gram

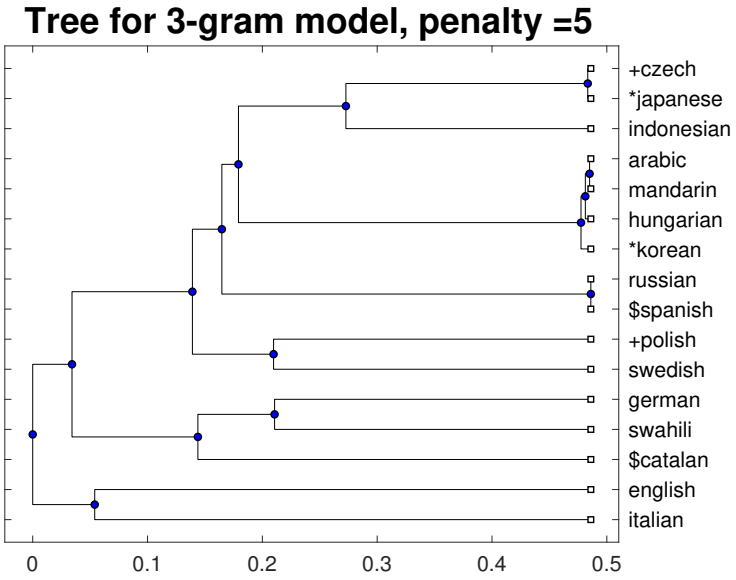


(b) Tree structure of tri-gram

Figure C.17: The 16 UNDHR text language distances results of tri-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 1. Figure C.17(a) shows the colormap of the language distance variations and Figure C.17(b) shows the language tree which is built by the distances. The colour variation in Figure C.17(a) shows the pairwise distances between languages.

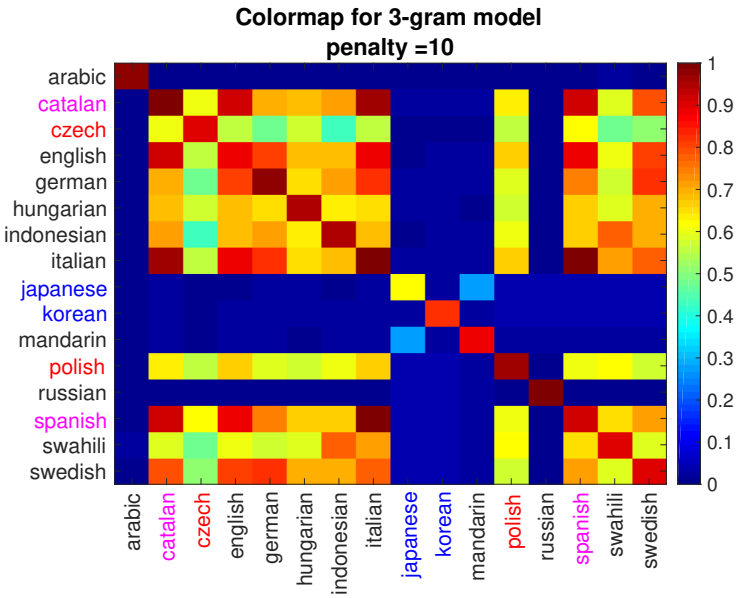


(a) Colormap of tri-gram

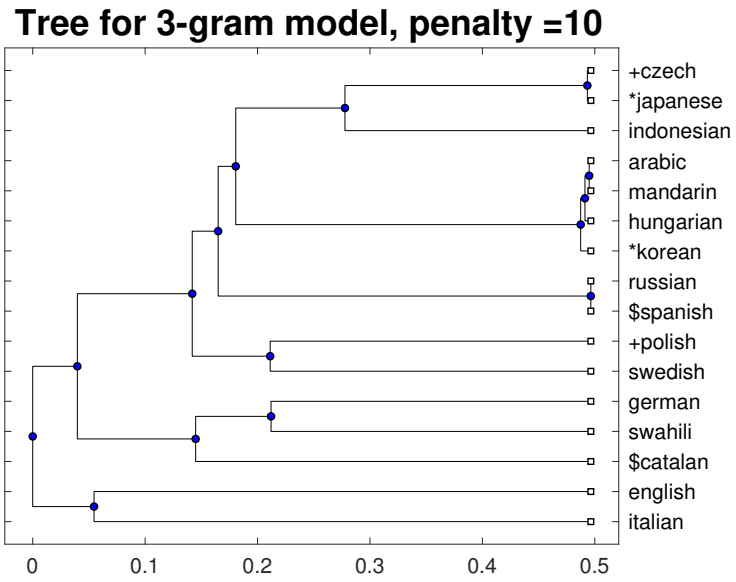


(b) Tree structure of tri-gram

Figure C.18: The 16 UNDHR text language distances results of tri-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 5. Figure C.18(a) shows the colormap of the language distance variations and Figure C.18(b) shows the language tree which is built by the distances. The colour variation in Figure C.18(a) shows the pairwise distances between languages.

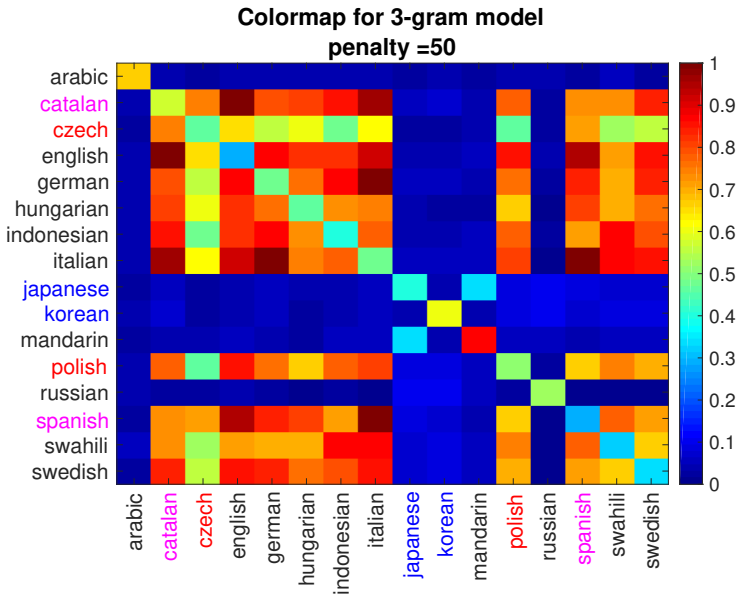


(a) Colormap of tri-gram

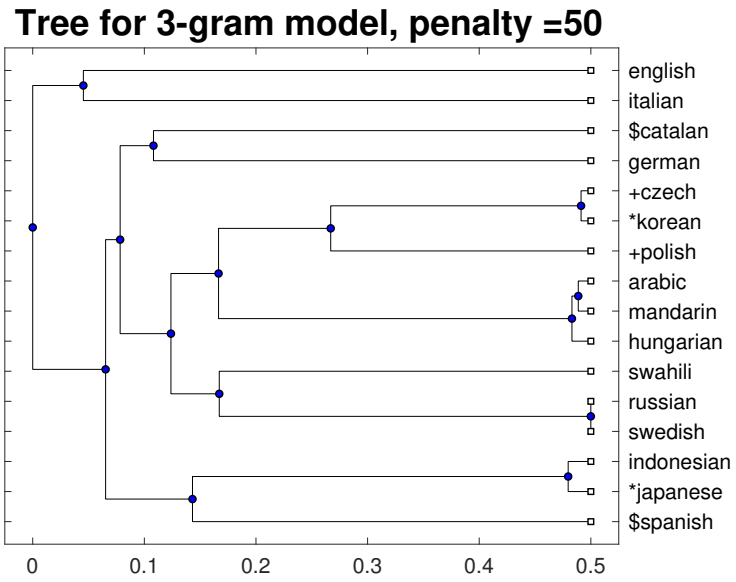


(b) Tree structure of tri-gram

Figure C.19: The 16 UNDHR text language distances results of tri-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 10. Figure C.19(a) shows the colormap of the language distance variations and Figure C.19(b) shows the language tree which is built by the distances. The colour variation in Figure C.19(a) shows the pairwise distances between languages.

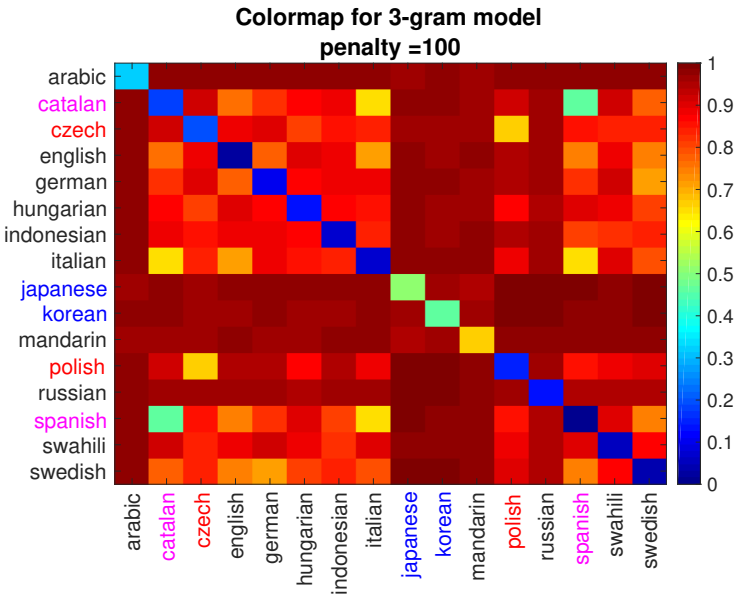


(a) Colormap of tri-gram

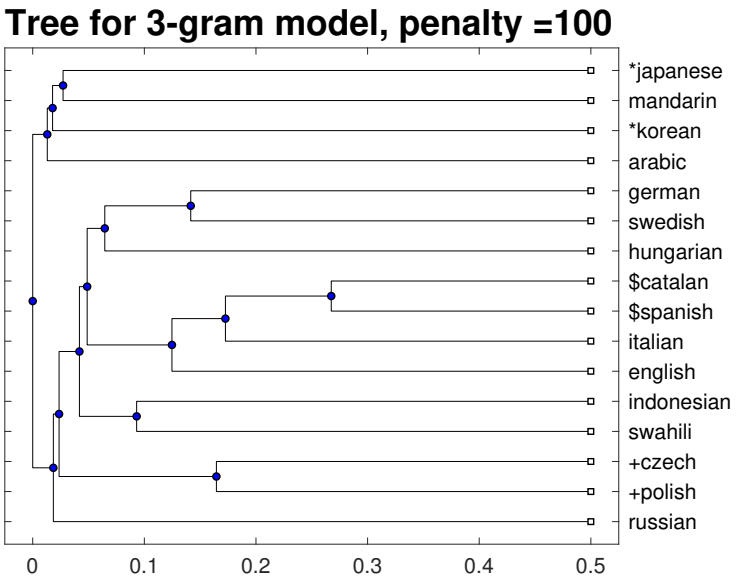


(b) Tree structure of tri-gram

Figure C.20: The 16 UNDHR text language distances results of tri-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 50. Figure C.20(a) shows the colormap of the language distance variations and Figure C.20(b) shows the language tree which is built by the distances. The colour variation in Figure C.20(a) shows the pairwise distances between languages.

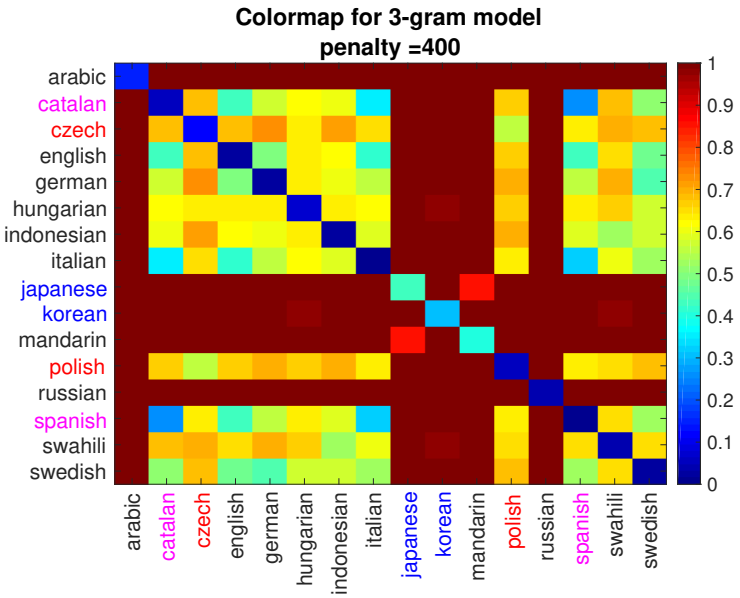


(a) Colormap of tri-gram

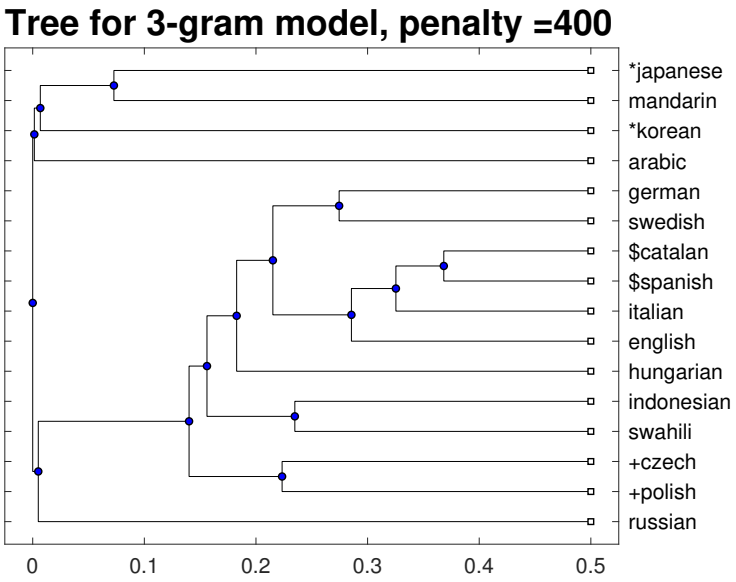


(b) Tree structure of tri-gram

Figure C.21: The 16 UNDHR text language distances results of tri-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 100. Figure C.21(a) shows the colormap of the language distance variations and Figure C.21(b) shows the language tree which is built by the distances. The colour variation in Figure C.21(a) shows the pairwise distances between languages.

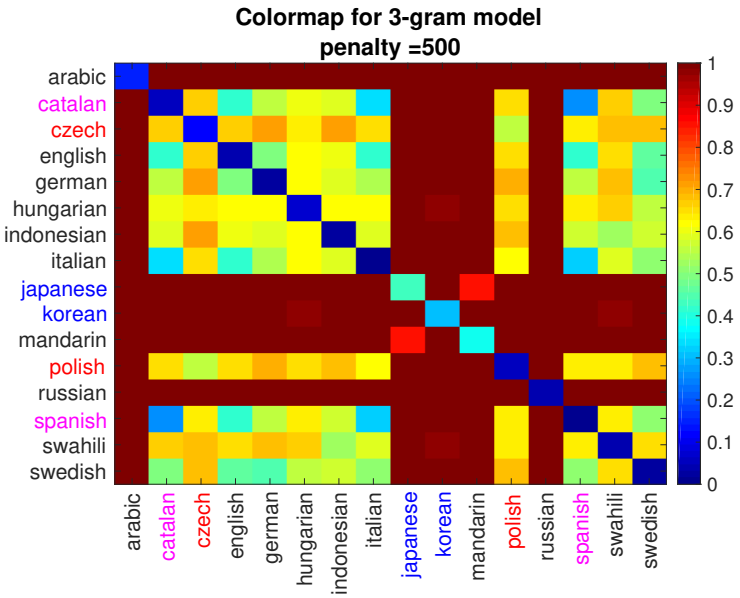


(a) Colormap of tri-gram

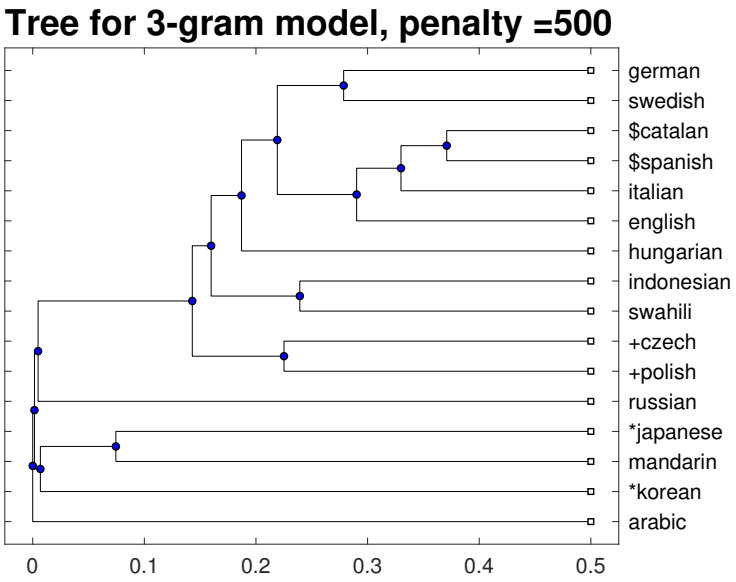


(b) Tree structure of tri-gram

Figure C.22: The 16 UNDHR text language distances results of tri-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 400. Figure C.22(a) shows the colormap of the language distance variations and Figure C.22(b) shows the language tree which is built by the distances. The colour variation in Figure C.22(a) shows the pairwise distances between languages.

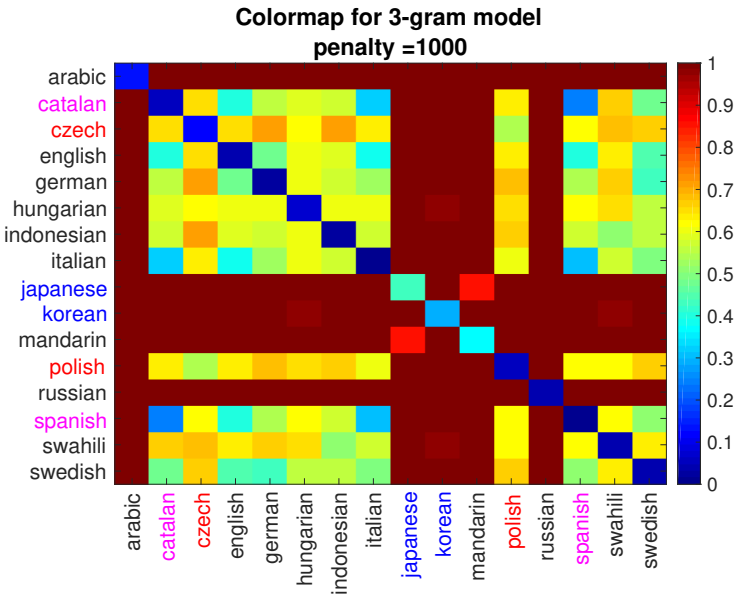


(a) Colormap of tri-gram

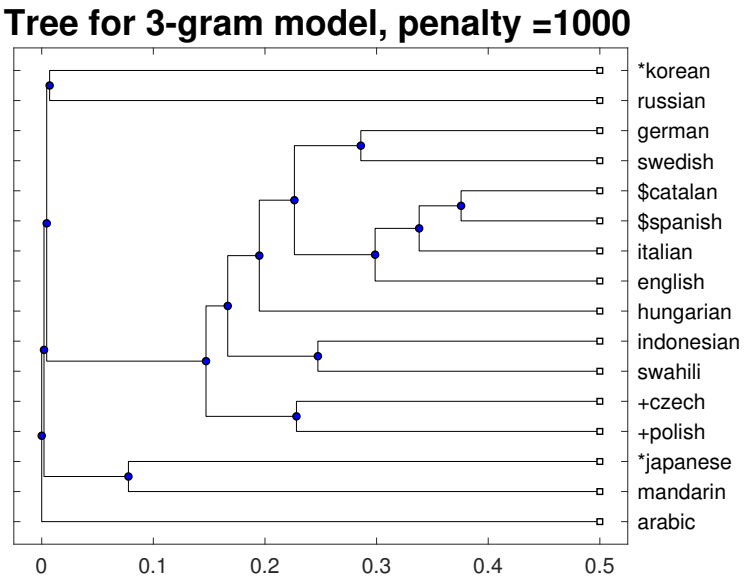


(b) Tree structure of tri-gram

Figure C.23: The 16 UNDHR text language distances results of tri-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 500. Figure C.23(a) shows the colormap of the language distance variations and Figure C.23(b) shows the language tree which is built by the distances. The colour variation in Figure C.23(a) shows the pairwise distances between languages.

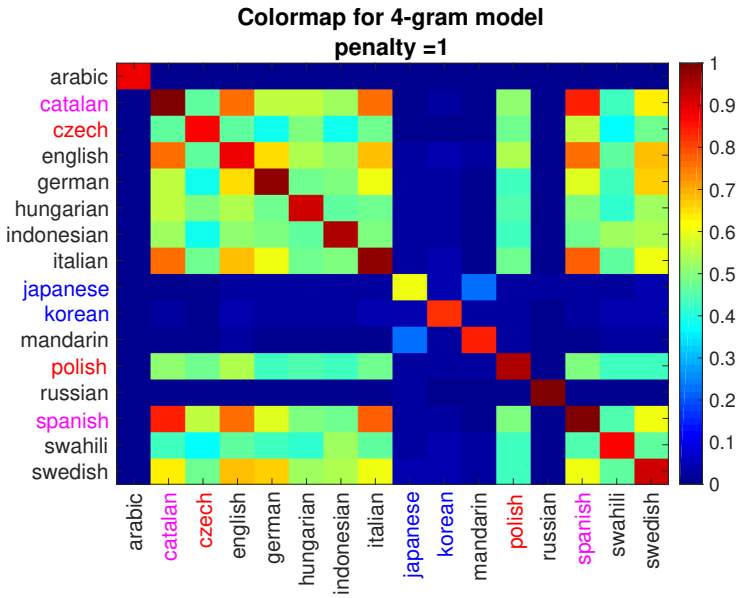


(a) Colormap of tri-gram

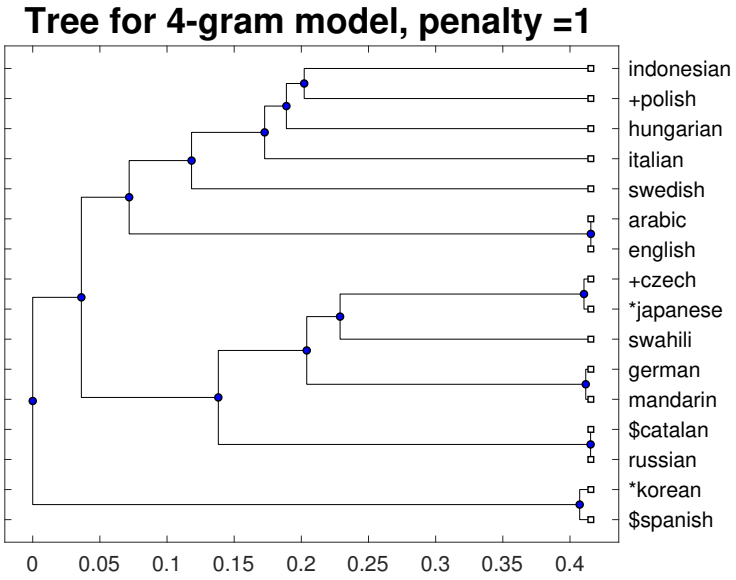


(b) Tree structure of tri-gram

Figure C.24: The 16 UNDHR text language distances results of tri-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 1000. Figure C.24(a) shows the colormap of the language distance variations and Figure C.24(b) shows the language tree which is built by the distances. The colour variation in Figure C.24(a) shows the pairwise distances between languages.

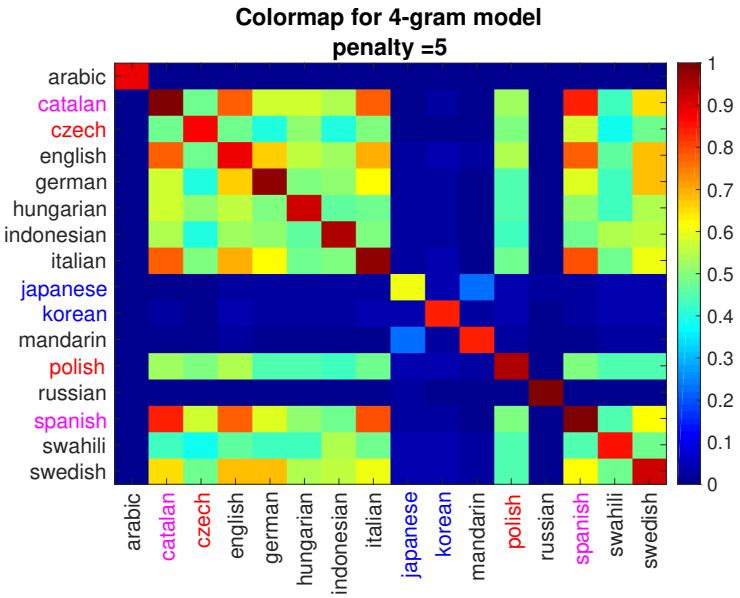


(a) Colormap of four-gram

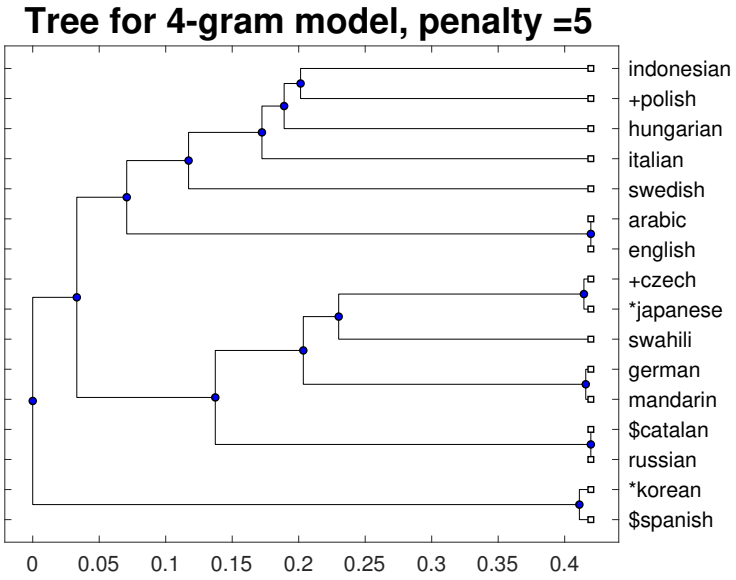


(b) Tree structure of four-gram

Figure C.25: The 16 UNDHR text language distances results of four-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 1. Figure C.25(a) shows the colormap of the language distance variations and Figure C.25(b) shows the language tree which is built by the distances. The colour variation in Figure C.25(a) shows the pairwise distances between languages.

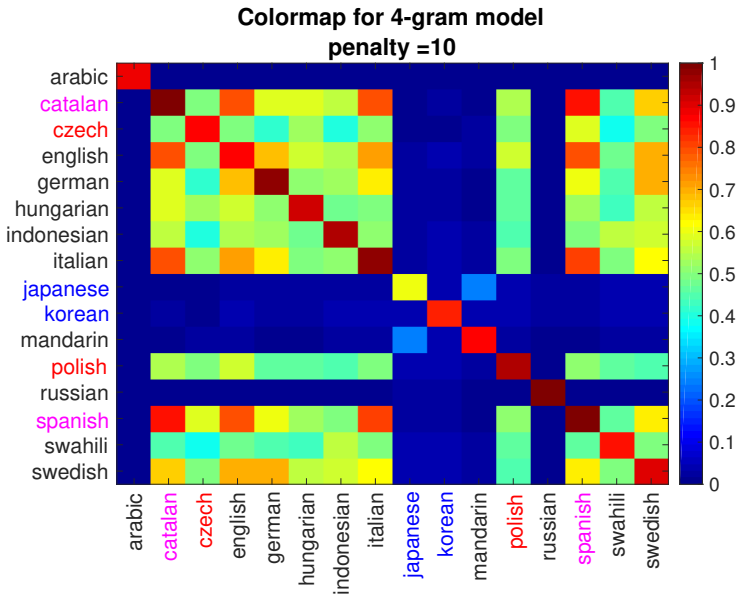


(a) Colormap of four-gram

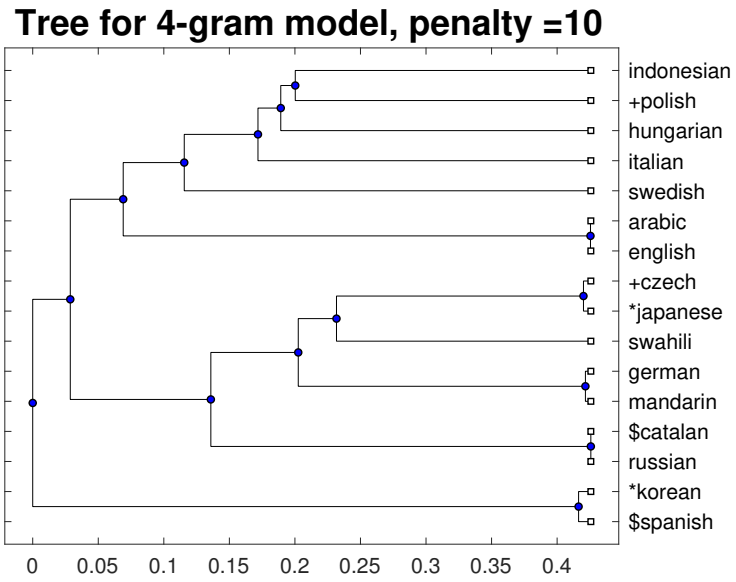


(b) Tree structure of four-gram

Figure C.26: The 16 UNDHR text language distances results of four-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 5. Figure C.26(a) shows the colormap of the language distance variations and Figure C.26(b) shows the language tree which is built by the distances. The colour variation in Figure C.26(a) shows the pairwise distances between languages.

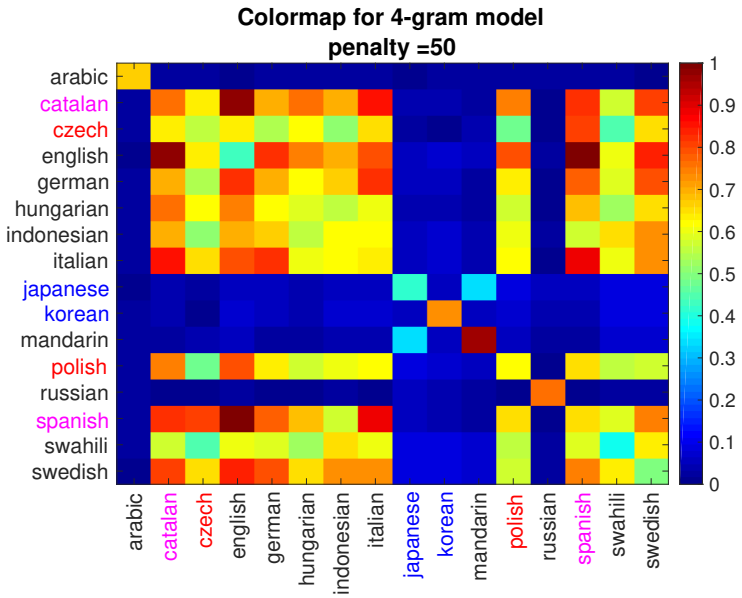


(a) Colormap of four-gram

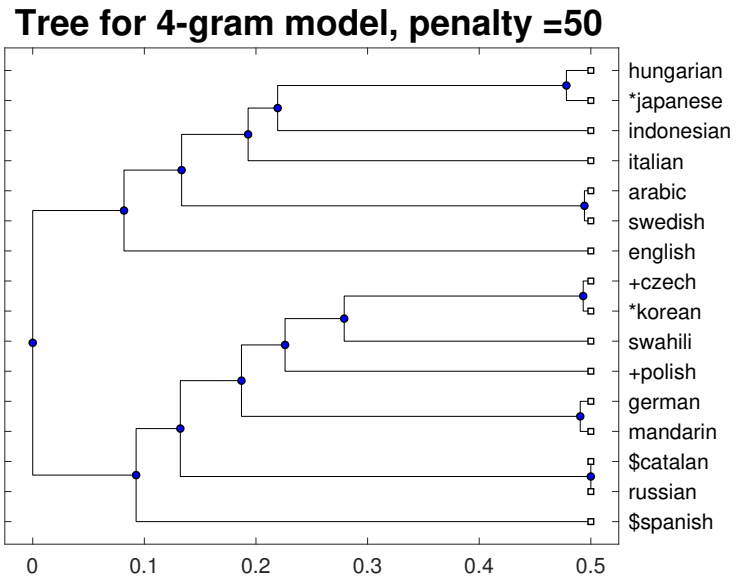


(b) Tree structure of four-gram

Figure C.27: The 16 UNDHR text language distances results of four-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 10. Figure C.27(a) shows the colormap of the language distance variations and Figure C.27(b) shows the language tree which is built by the distances. The colour variation in Figure C.27(a) shows the pairwise distances between languages.

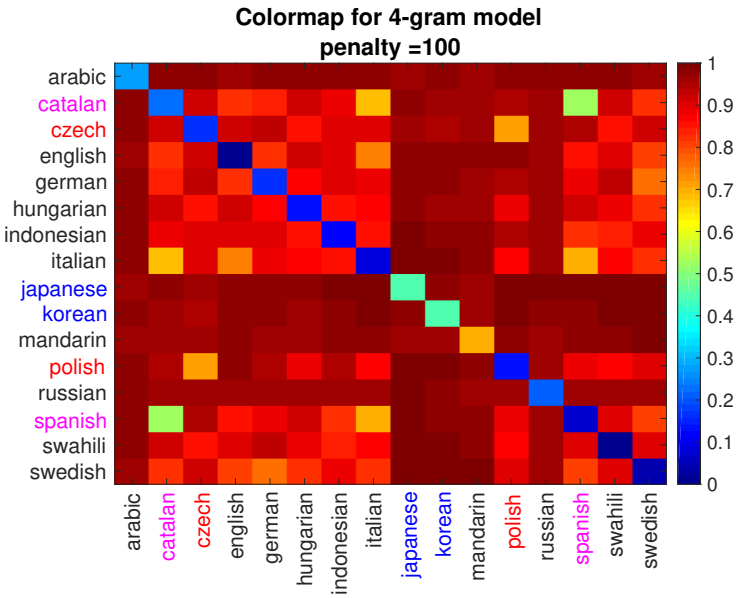


(a) Colormap of four-gram

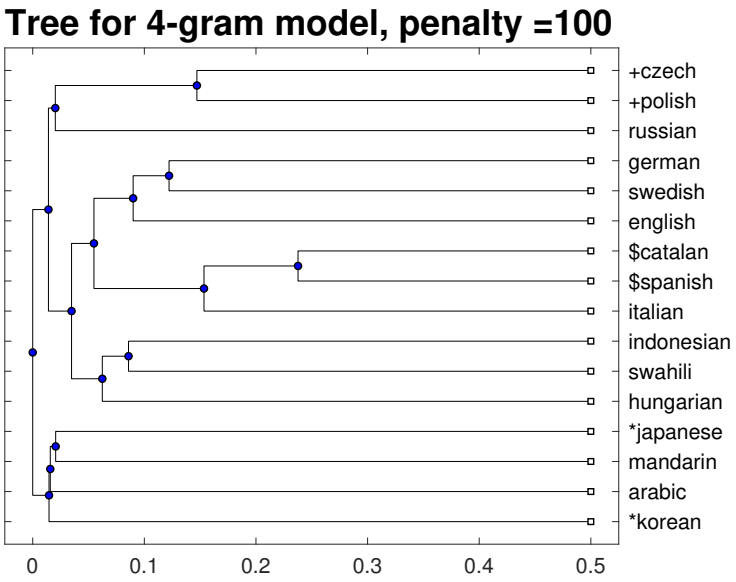


(b) Tree structure of four-gram

Figure C.28: The 16 UNDHR text language distances results of four-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 50. Figure C.28(a) shows the colormap of the language distance variations and Figure C.28(b) shows the language tree which is built by the distances. The colour variation in Figure C.28(a) shows the pairwise distances between languages.

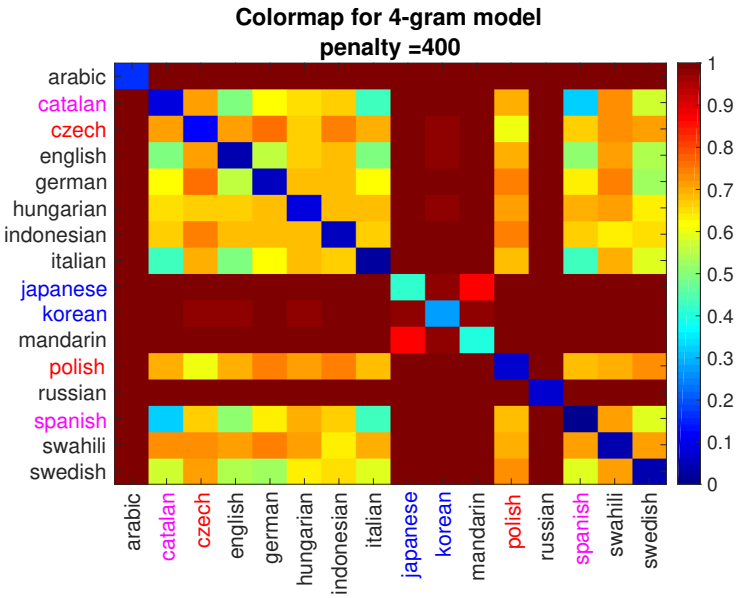


(a) Colormap of four-gram

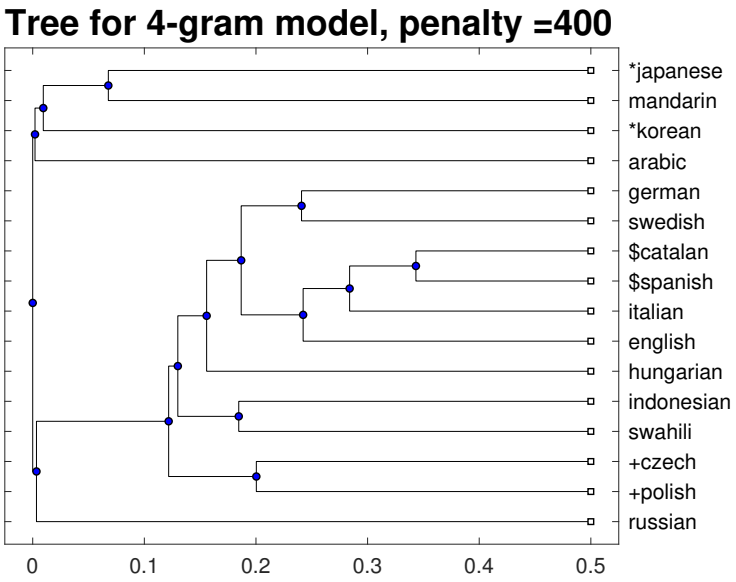


(b) Tree structure of four-gram

Figure C.29: The 16 UNDHR text language distances results of four-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 100. Figure C.29(a) shows the colormap of the language distance variations and Figure C.29(b) shows the language tree which is built by the distances. The colour variation in Figure C.29(a) shows the pairwise distances between languages.

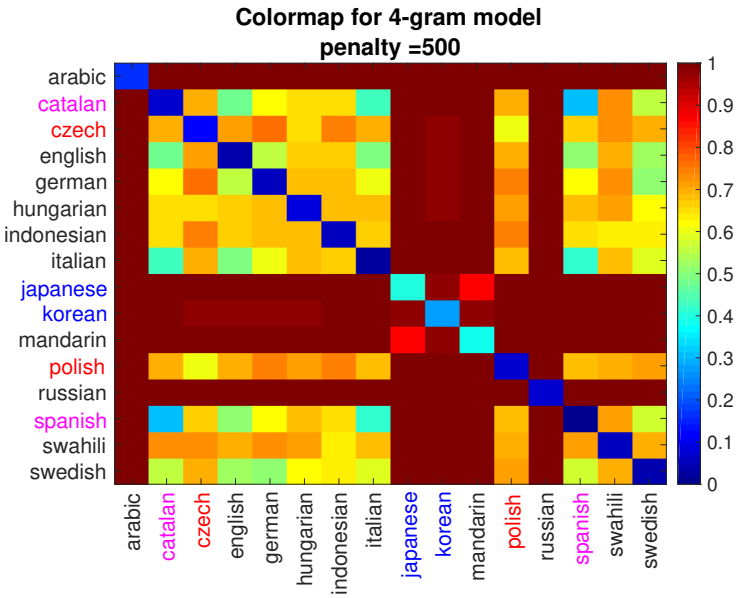


(a) Colormap of four-gram

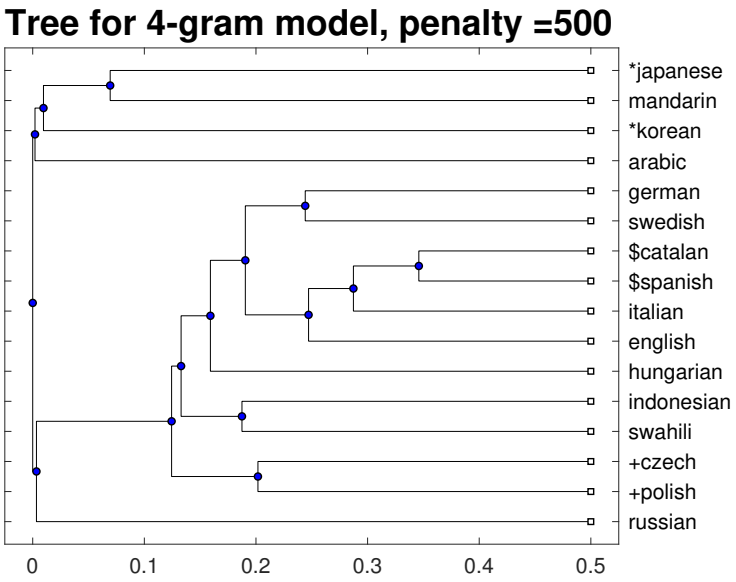


(b) Tree structure of four-gram

Figure C.30: The 16 UNDHR text language distances results of four-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 400. Figure C.30(a) shows the colormap of the language distance variations and Figure C.30(b) shows the language tree which is built by the distances. The colour variation in Figure C.30(a) shows the pairwise distances between languages.

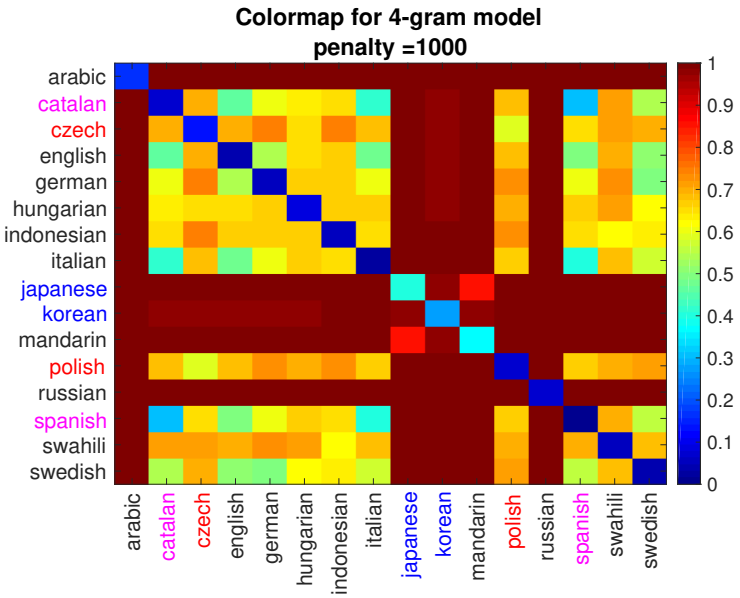


(a) Colormap of four-gram

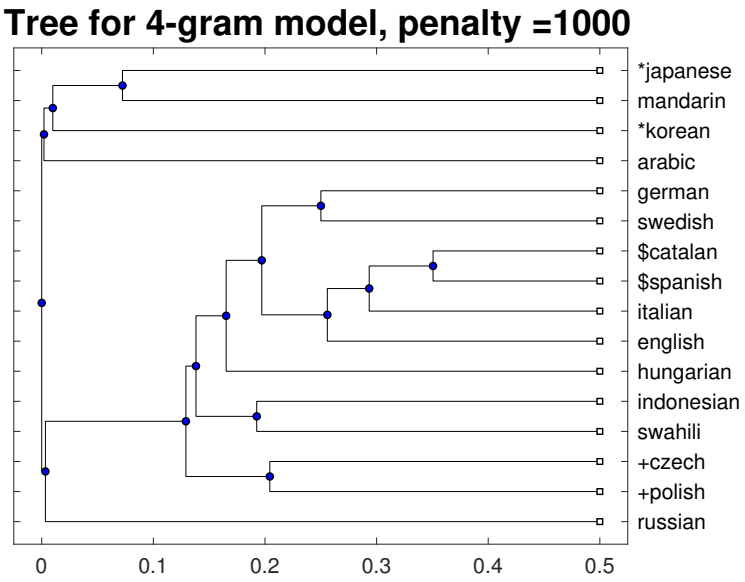


(b) Tree structure of four-gram

Figure C.31: The 16 UNDHR text language distances results of four-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 500. Figure C.31(a) shows the colormap of the language distance variations and Figure C.31(b) shows the language tree which is built by the distances. The colour variation in Figure C.31(a) shows the pairwise distances between languages.

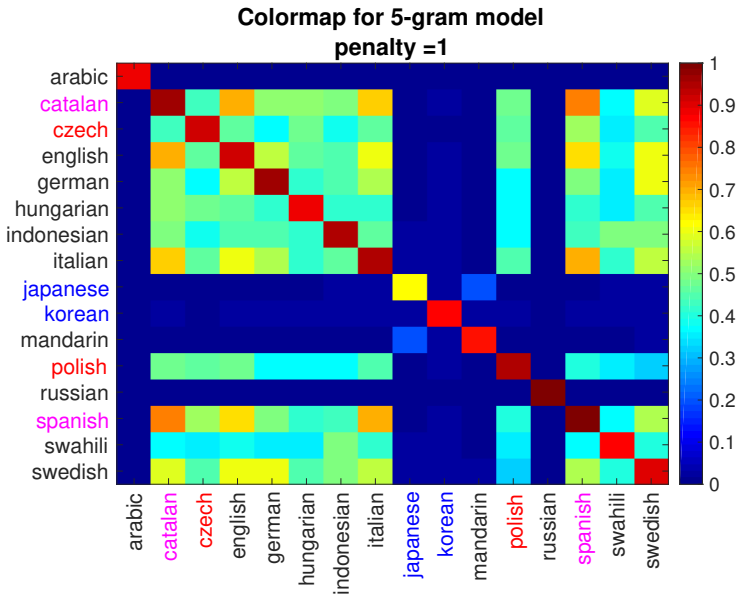


(a) Colormap of four-gram

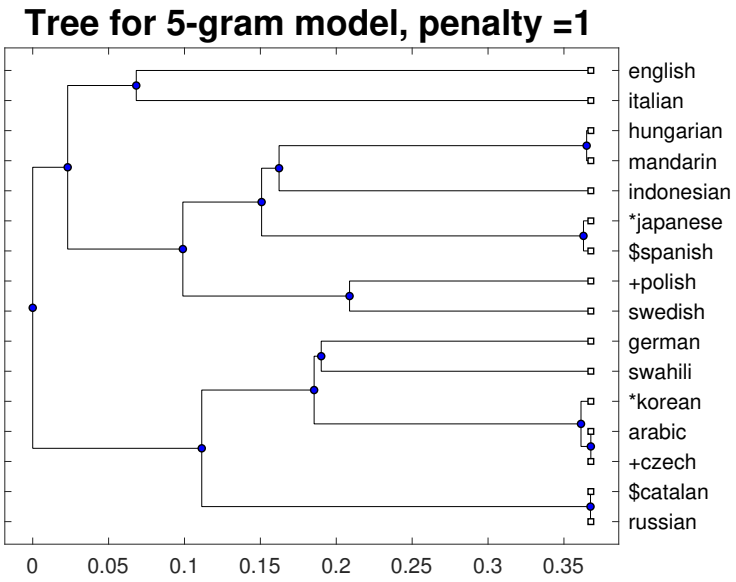


(b) Tree structure of four-gram

Figure C.32: The 16 UNDHR text language distances results of four-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 1000. Figure C.32(a) shows the colormap of the language distance variations and Figure C.32(b) shows the language tree which is built by the distances. The colour variation in Figure C.32(a) shows the pairwise distances between languages.

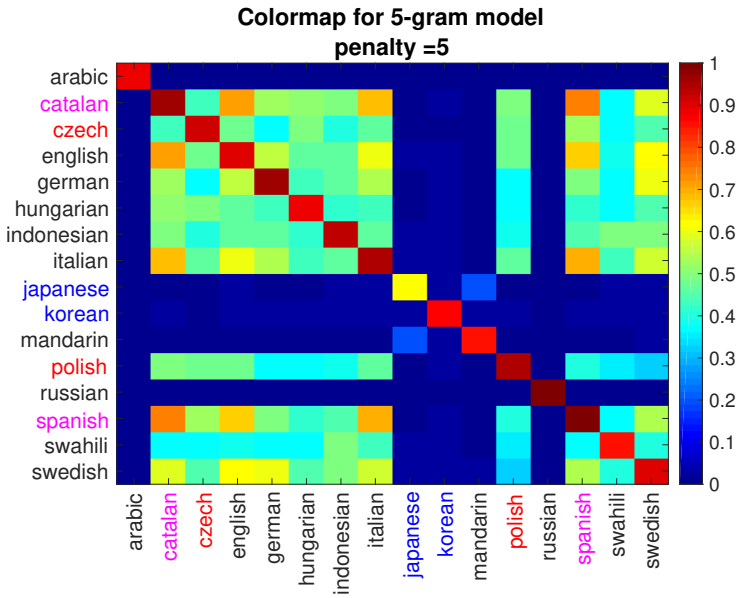


(a) Colormap of five-gram

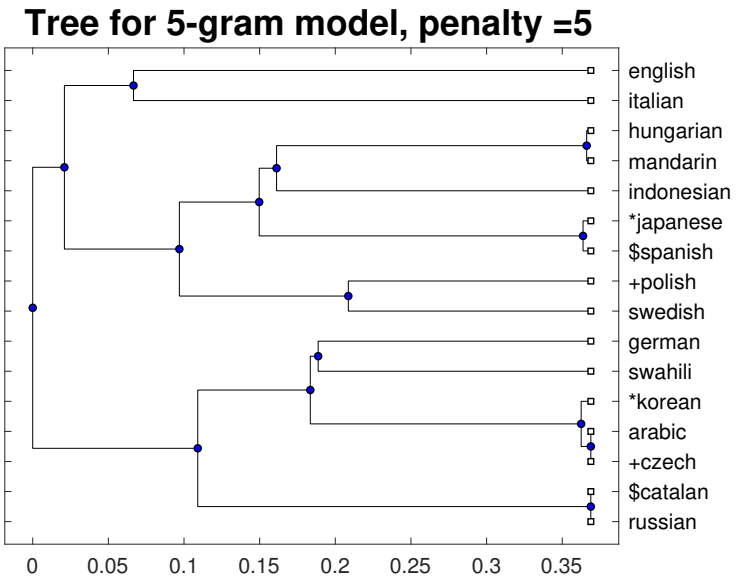


(b) Tree structure of five-gram

Figure C.33: The 16 UNDHR text language distances results of five-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 1. Figure C.33(a) shows the colormap of the language distance variations and Figure C.33(b) shows the language tree which is built by the distances. The colour variation in Figure C.33(a) shows the pairwise distances between languages.

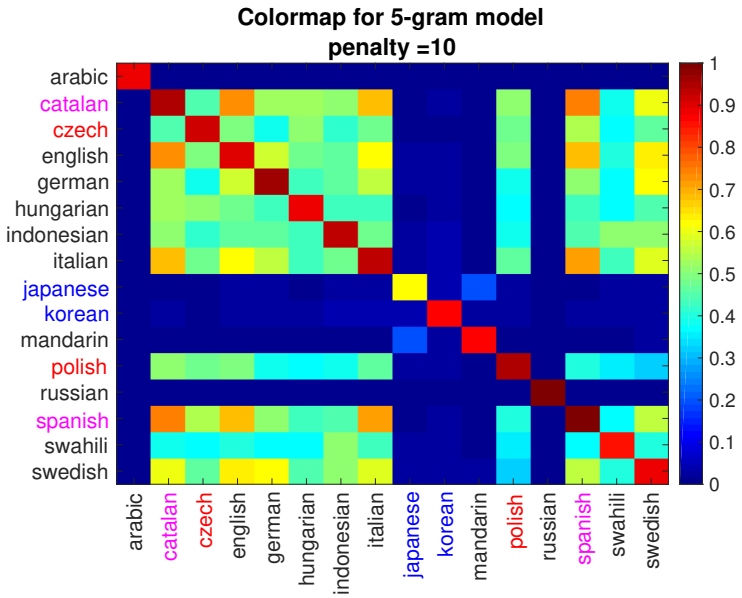


(a) Colormap of five-gram

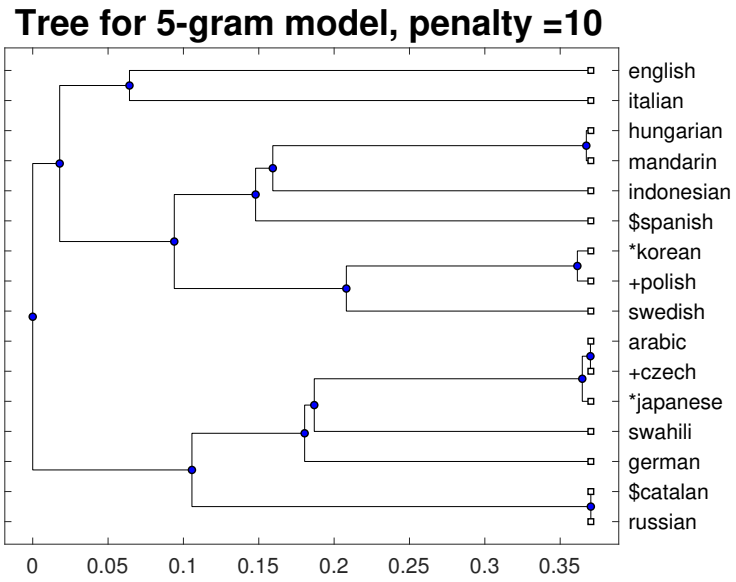


(b) Tree structure of five-gram

Figure C.34: The 16 UNDHR text language distances results of five-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 5. Figure C.34(a) shows the colormap of the language distance variations and Figure C.34(b) shows the language tree which is built by the distances. The colour variation in Figure C.34(a) shows the pairwise distances between languages.

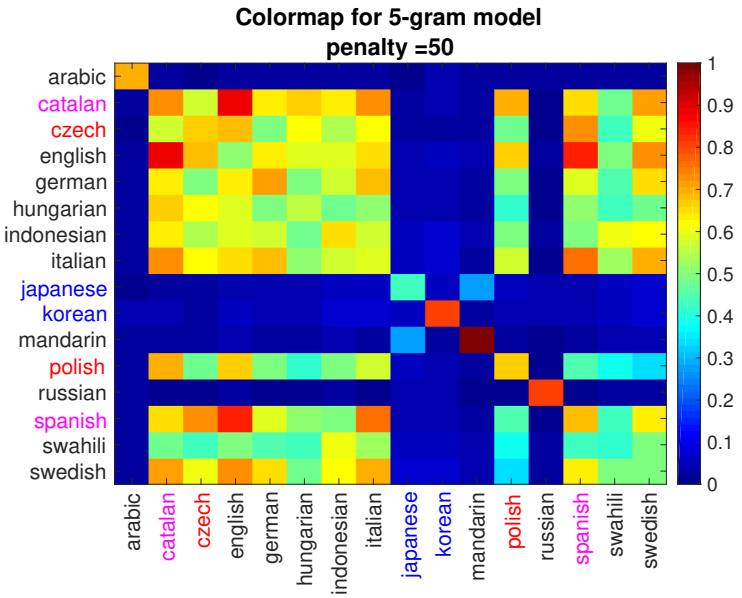


(a) Colormap of five-gram

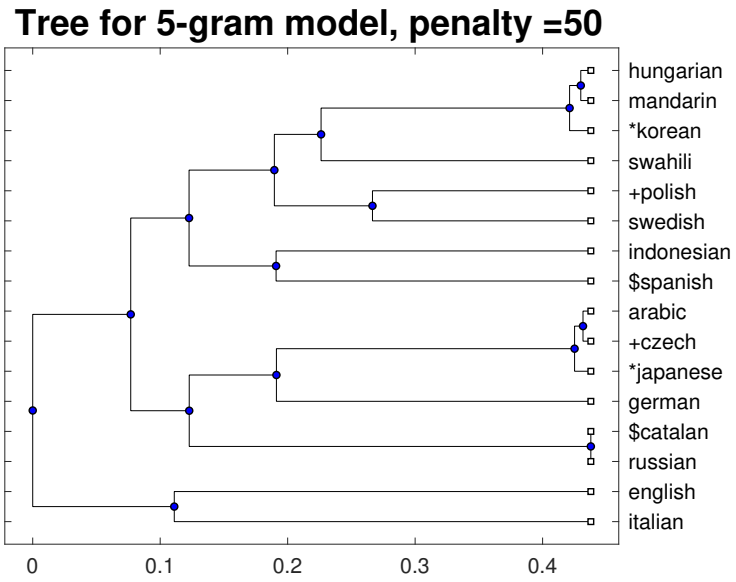


(b) Tree structure of five-gram

Figure C.35: The 16 UNDHR text language distances results of five-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 10. Figure C.35(a) shows the colormap of the language distance variations and Figure C.35(b) shows the language tree which is built by the distances. The colour variation in Figure C.35(a) shows the pairwise distances between languages.

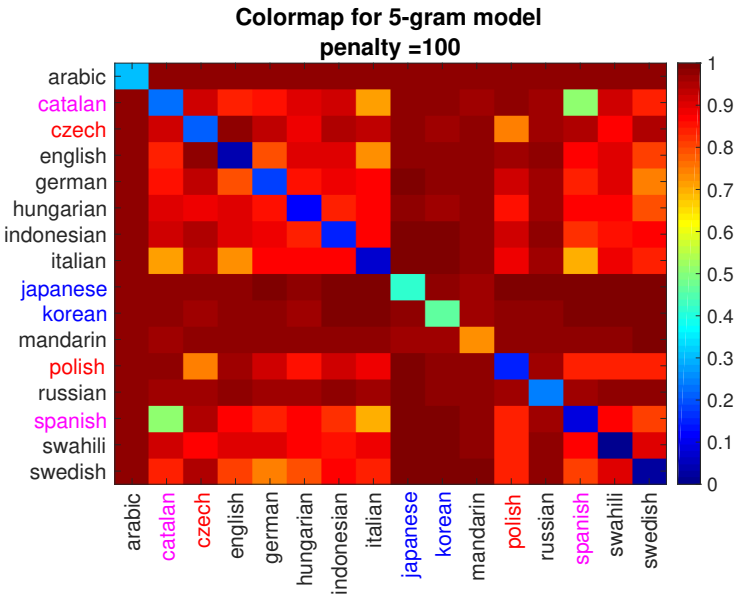


(a) Colormap of five-gram

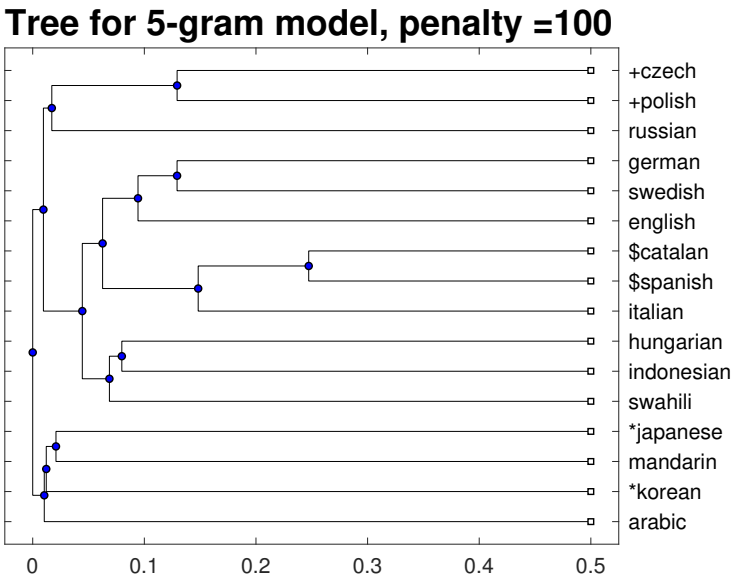


(b) Tree structure of five-gram

Figure C.36: The 16 UNDHR text language distances results of five-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 50. Figure C.36(a) shows the colormap of the language distance variations and Figure C.36(b) shows the language tree which is built by the distances. The colour variation in Figure C.36(a) shows the pairwise distances between languages.

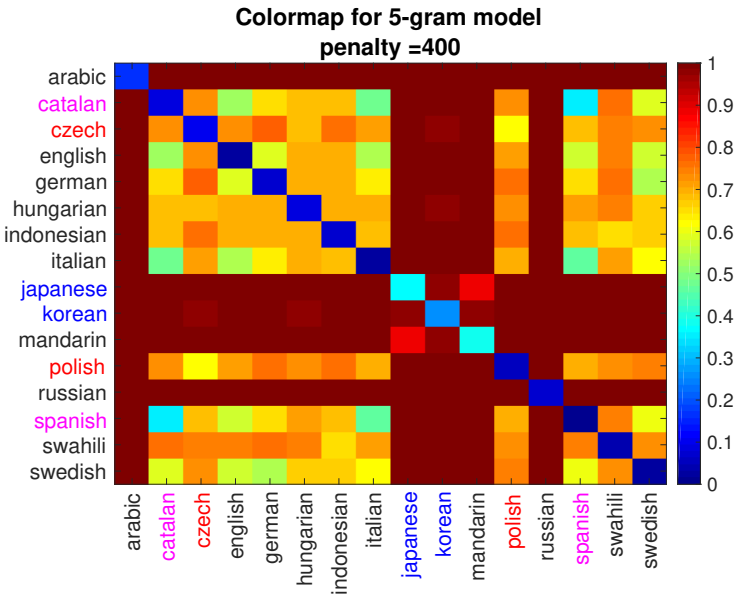


(a) Colormap of five-gram

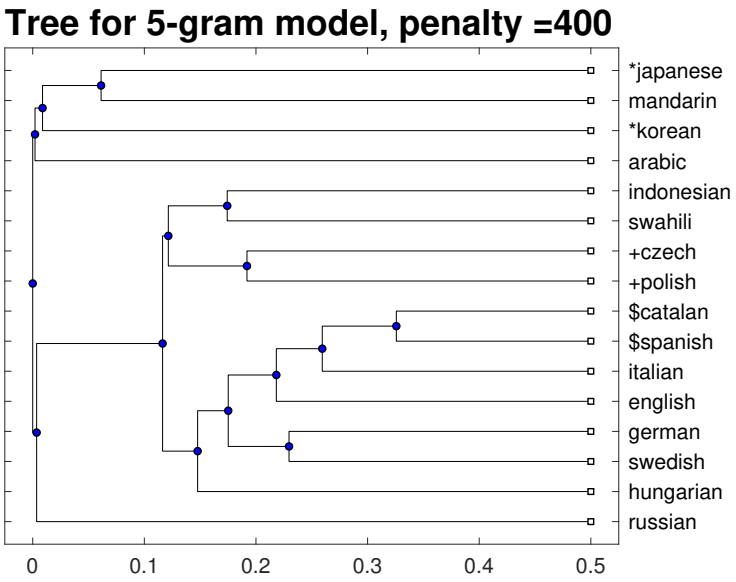


(b) Tree structure of five-gram

Figure C.37: The 16 UNDHR text language distances results of five-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 100. Figure C.37(a) shows the colormap of the language distance variations and Figure C.37(b) shows the language tree which is built by the distances. The colour variation in Figure C.37(a) shows the pairwise distances between languages.

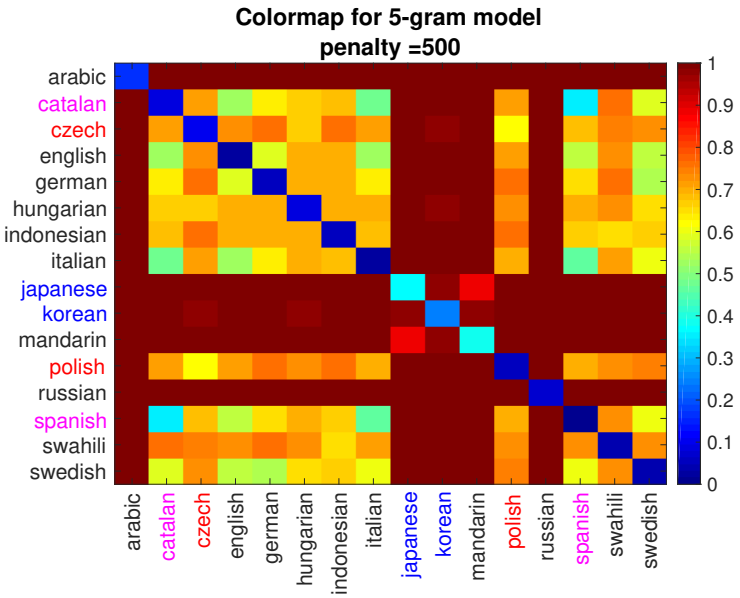


(a) Colormap of five-gram

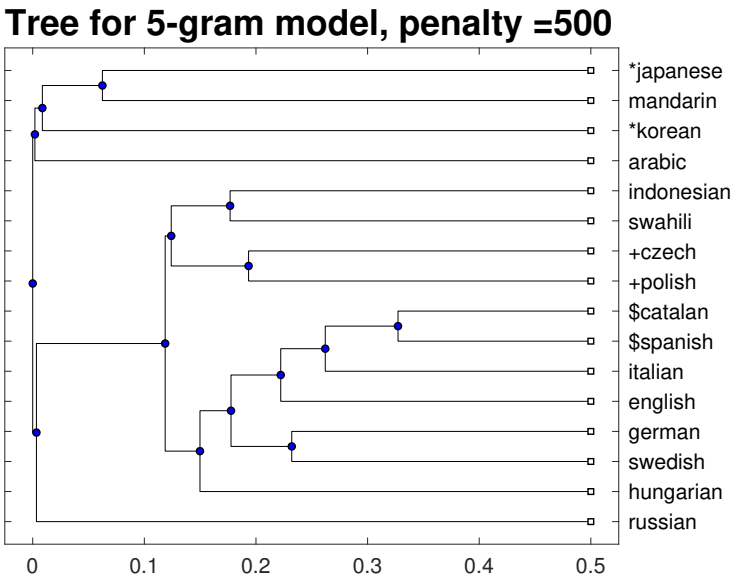


(b) Tree structure of five-gram

Figure C.38: The 16 UNDHR text language distances results of five-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 400. Figure C.38(a) shows the colormap of the language distance variations and Figure C.38(b) shows the language tree which is built by the distances. The colour variation in Figure C.38(a) shows the pairwise distances between languages.

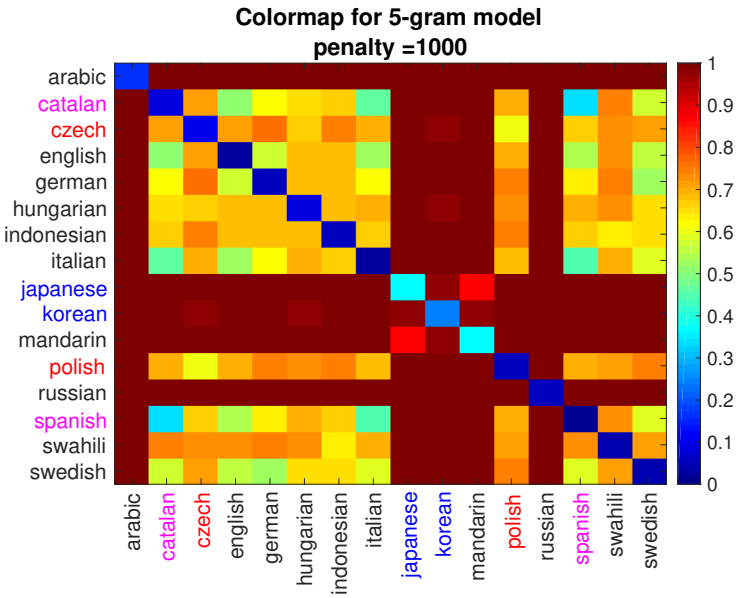


(a) Colormap of five-gram

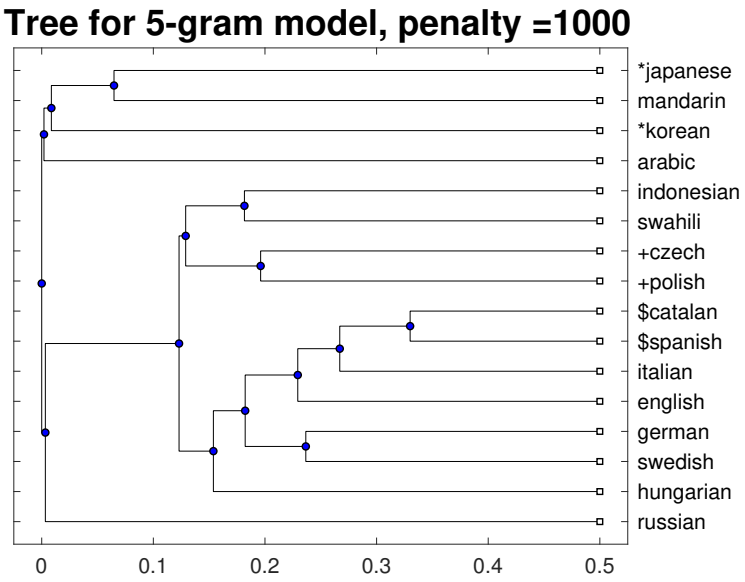


(b) Tree structure of five-gram

Figure C.39: The 16 UNDHR text language distances results of five-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 500. Figure C.39(a) shows the colormap of the language distance variations and Figure C.39(b) shows the language tree which is built by the distances. The colour variation in Figure C.39(a) shows the pairwise distances between languages.



(a) Colormap of five-gram



(b) Tree structure of five-gram

Figure C.40: The 16 UNDHR text language distances results of five-gram. The distances shown in the diagrams are $distance/\sigma$ and are normalized into $[0, 1]$. The penalty value is 1000. Figure C.40(a) shows the colormap of the language distance variations and Figure C.40(b) shows the language tree which is built by the distances. The colour variation in Figure C.40(a) shows the pairwise distances between languages.

References

Aguilar-Torres, G., Toscano-Medina, K., Sanchez-Perez, G., Nakano-Miyatake, M., and Perez-Meana, H. (2009). Eigenface-Gabor algorithm for feature extraction in face recognition. *International Journal of Computers*, 3:20–30.

Ahmed, N., Natarajan, T., and Rao, K. (1974). Discrete Cosine Transform. *Computers, IEEE Transactions on*, 100(1):90–93.

Aliprand, J. M. (2011). The unicode standard. *Library resources & technical services*, 44(3):160–167.

Ambikairajah, E., Li, H., Wang, L., Yin, B., and Sethu, V. (2011). Language identification: a tutorial. *Circuits and Systems Magazine, IEEE*, 11(2):82–108.

Association, I. P. et al. (1999). *Handbook of the International Phonetic Association: A guide to the use of the International Phonetic Alphabet*. Cambridge University Press.

Atal, B. S. (1976). Automatic recognition of speakers from their voices. *Proceedings of the IEEE*, 64(4):460–475.

Baum, L. (1972). An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes. *Inequalities*, 3:1–8.

Beesley, K. R. (1988). Language identifier: A computer program for automatic natural-language identification of on-line text. In *Proceedings of the 29th Annual Conference of the American Translators Association*, volume 47, page 54. Citeseer.

Benedetto, D., Caglioti, E., and Loreto, V. (2002). Language trees and zipping. *Physical Review Letters*, 88(4):048702.

Bielefeld, B. (1994). Language identification using shifted delta cepstrum. In *Fourteenth Annual Speech Research Symposium*.

Bray, T., Paoli, J., M Sperberg-McQueen, C., Maler, E., and Yergeau, F. (2008). Extensible markup language (xml) 1.0 (fifth edition).

Bregler, C. and Konig, Y. (2002). "Eigenlips" for robust speech recognition. In *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, volume 2. IEEE.

Burrows, Michael, W. D. J. (1994). A block-sorting lossless data compression algorithm. Includes bibliographical references.

Campana, B. J. and Keogh, E. J. (2010). A compression-based distance measure for texture. *Statistical Analysis and Data Mining*, 3(6):381–398.

Cavnar, W. B. and Trenkle, J. M. (1994). N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, US.

Cetingul, H., Yemez, Y., Erzin, E., and Tekalp, A. (2006). Discriminative analysis of lip motion features for speaker identification and speech-reading. *Image Processing, IEEE Transactions on*, 15(10):2879–2891.

Chopra, V., Eaves, J., Jones, R., Li, S., and Bell, J. T. (2005). *Beginning JavaServer Pages*. John Wiley & Sons.

Cilibrasi, R. and Vitányi, P. M. (2005). Clustering by compression. *Information Theory, IEEE Transactions on*, 51(4):1523–1545.

Cleary, J. G. and Witten, I. (1984). Data compression using adaptive coding and partial string matching. *Communications, IEEE Transactions on*, 32(4):396–402.

Clive, S., Churcher, G., Hayes, J., Hughes, J., and Johnson, S. (1994). Natural language identification using corpus-based models. *Hermes Journal of Linguistics*, 13(S 183):203.

Collinge, N. E. (2002). *An encyclopedia of language*. Routledge.

Cootes, T., Edwards, G., and Taylor, C. (2001a). Active appearance models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(6):681–685.

Cootes, T., Taylor, C., and Lanitis, A. (1994). Multi-resolution search with active shape models. In *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, volume 1, pages 610–612. IEEE.

Cootes, T. F., Edwards, G. J., and Taylor, C. J. (2001b). Active appearance models. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):681–685.

De La Torre, A., Segura, J. C., Benitez, C., Peinado, A. M., and Rubio, A. J. (2002). Non-linear transformations of the feature space for robust speech recognition. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 1, pages I–401. IEEE.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38.

Deutsch, L. P. (1996). DEFLATE Compressed Data Format Specification version 1.3. RFC 1951.

Dunning, T. (1994). *Statistical identification of language*. Computing Research Laboratory, New Mexico State University.

Freedman, D. and Diaconis, P. (1981). On the histogram as a density estimator: L 2 theory. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 57(4):453–476.

Gold, B., Morgan, N., and Ellis, D. (2011). *Speech and audio signal processing: processing and perception of speech and music*. Wiley. com.

Gold, E. M. (1967). Language identification in the limit. *Information and control*, 10(5):447–474.

Goodman, J. (2002). Extended comment on language trees and zipping. *arXiv preprint cond-mat/0202383*.

Gray, R. (1984). Vector quantization. *IEEE Assp Magazine*, 1(2):4–29.

Hao, Y., Campana, B., and Keogh, E. (2012). Monitoring and mining insect sounds in visual space. In *Proceedings of the 2012 SIAM International Conference on Data Mining*, pages 792–803. SIAM.

Henrich, P. (1989). Language identification for the automatic grapheme-to-phoneme conversion of foreign words in a german text-to-speech system. In *EUROSPEECH*, pages 2220–2223. ISCA.

Hochberg, J., Bowers, K., Cannon, M., and Kelly, P. (1999). Script and language identification for handwritten document images. *International Journal on Document Analysis and Recognition*, 2(2-3):45–52.

Huffman, D. A. et al. (1952). A method for the construction of minimum redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101.

Hughes, B., Baldwin, T., Bird, S., Nicholson, J., and MacKinlay, A. (2006). Reconsidering language identification for written language resources. In *Proc. International Conference on Language Resources and Evaluation*, pages 485–488.

Ingle, N. C. (1976). A language identification table. *The Incorporated Linguist*, 15(4):98–101.

International Phonetic Association (2018). *The International Phonetic Alphabet (revised to 2018)*. International Phonetic Association. bibtex: international_phonetic_association_international_2018.

Kass, M., Witkin, A., and Terzopoulos, D. (1988). Snakes: Active contour models. *International journal of computer vision*, 1(4):321–331.

Kučera, H. and Monroe, G. K. (1968). *A comparative quantitative phonology of Russian, Czech, and German*. Number 4. Elsevier.

Laver, J. (1994). *Principles of phonetics*. Cambridge University Press.

Lee, S. and Hasegawa, T. (2011). Bayesian phylogenetic analysis supports an agricultural origin of japonic languages. *Proceedings of the Royal Society B: Biological Sciences*, 278(1725):3662–3669.

Lewis, M. Paul, G. F. S. and D.Fenning, C. (2013). Ethnologue: Languages of the world, seventeenth edition. *Dallas, Texas: SIL International*.

Lu, B., Zhang, L., and Leong, H. W. (2017). A program to compute the soft robinson–foulds distance between phylogenetic networks. *BMC genomics*, 18(2):111.

Manning, C. D. and Schütze, H. (1999). *Foundations of statistical natural language processing*, volume 999. MIT Press.

Markel, J. D. and Gray, A. J. (2013). *Linear prediction of speech*, volume 12. Springer Science & Business Media.

Matthews, I. (1998). *Feature for Audio-Visual Speech Recognition*. PhD thesis, School of Information System, University of East Anglia, UK.

Matthews, I., Cootes, T., Bangham, J., Cox, S., and Harvey, R. (2002). Extraction of visual features for lipreading. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(2):198–213.

Matthews, I., Cootes, T., Cox, S., Harvey, R., and Bangham, J. (1998). Lipreading using shape, shading and scale. *Proceedings-institute of Acoustics*, 20:99–106.

Mustonen, S. (1965). Multiple discriminant analysis in linguistic problems. *Statistical Methods in Linguistics*, 4:37–44.

Newman, J. L. (2011). *Language Identification Using Visual Features*. PhD thesis, School of Information System, University of East Anglia, UK.

Parzen, E. (1962). On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076.

Peake, G. and Tan, T. (1997). Script and language identification from document images. In *Proceedings Workshop on Document Image Analysis (DIA'97)*, pages 10–17. IEEE.

Pekalska, E., de Ridder, D., Duin, R. P., and Kraaijveld, M. A. (1999). A new method of generalizing sammon mapping with application to algorithm speed-up. In *ASCI*, volume 99, pages 221–228.

Pelecanos, J. and Sridharan, S. (2001). Feature warping for robust speaker verification. In *2001: A Speaker Odyssey - The Speaker Recognition Workshop*, pages 213–218, Crete, Greece. International Speech Communication Association (ISCA).

Poutsma, A. (2002). Applying monte carlo techniques to language identification. *Language and Computers*, 45(1):179–189.

Rabiner, L. R. and Juang, B. (1993). *Fundamentals of speech recognition*. Prentice Hall signal processing series. Prentice Hall.

Raggett, D., Le Hors, A., Jacobs, I., et al. (1999). Html 4.01 specification. *W3C recommendation*, 24.

Robinson, A. and Cherry, C. (1967). Results of a prototype television bandwidth compression scheme. *Proceedings of the IEEE*, 55(3):356–364.

Robinson, D. F. and Foulds, L. R. (1981). Comparison of phylogenetic trees. *Mathematical biosciences*, 53(1):131–147.

Ruhlen, M. (1991). *A guide to the world's languages: classification*, volume 1. Stanford University Press.

Ryabko, B. Y. (1980). Data compression by means of a "book stack". *Problemy Peredachi Informatsii*, 16(4):16–21.

Sammon, J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on computers*, 18(5):401–409.

Schilling, R. J. and Harris, S. L. (2012). *Introduction to digital signal processing using MATLAB*. Cengage Learning.

Scott, D. W. (1979). On optimal and data-based histograms. *Biometrika*, 66(3):605–610.

Seward, J. (1996). bzip2 and libbzip2. *avaliable at http://www. bzip. org*.

Shannon, C. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55.

Shepard, D. (1968). A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, pages 517–524. ACM.

Sibun, P. and Reynar, J. C. (1996). Language identification: Examining the issues. In *5th Symposium on Document Analysis and Information Retrieval*, pages 125–135, Las Vegas, Nevada, U.S.A.

Sircombe, K. (2000). *The usefulness and limitations of binned frequency histograms and probability density distributions for displaying absolute age data*. Natural Resources Canada, Geological Survey of Canada.

Starner, T. E. (1995). Visual recognition of american sign language using hidden markov models. Technical report, DTIC Document.

Stevens, S. S., Volkmann, J., and Newman, E. B. (1937). A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190.

Sturges, H. A. (1926). The choice of a class interval. *Journal of the american statistical association*, 21(153):65–66.

Sumby, W. H. and Pollack, I. (1954). Visual contribution to speech intelligibility in noise. *The journal of the acoustical society of america*, 26(2):212–215.

Tamura, H., Mori, S., and Yamawaki, T. (1978). Textural features corresponding to visual perception. *IEEE Transactions on Systems, man, and cybernetics*, 8(6):460–473.

The Unicode Consortium (2011). The Unicode Standard. Technical Report Version 6.0.0, Unicode Consortium, Mountain View, CA.

Tong, R., Ma, B., Zhu, D., Li, H., and Chng, E. S. (2006). Integrating acoustic, prosodic and phonotactic features for spoken language identification. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 1, pages I–I. IEEE.

Tsvetovat, M. and Kouznetsov, A. (2011). *Social Network Analysis for Startups: Finding connections on the social web*. " O'Reilly Media, Inc.".

Turk, M. and Pentland, A. (1991). Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86.

Unicode, I. (2013). Unicode 6.2 character code charts.

Voegelin, C. F. and Voegelin, F. M. (1977). *Classification and Index of the World Languages*. Elsevier, Amsterdam.

Witten, I. H., Neal, R. M., and Cleary, J. G. (1987). Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540.

Wolf, M., Whistler, K., Wicksteed, C., Davis, M., and Freytag, A. (2000). A standard compression scheme for unicode. *Unicode Tech.*

Young, S. J., Kershaw, D., Odell, J., Ollason, D., Valtchev, V., and Woodland, P. (2006). *The HTK Book Version 3.4*. Cambridge University Press.

Zipf, G. K. (1949). Human behavior and the principle of least effort.

Zissman, M. A. (1996). Comparison of four approaches to automatic language identification of telephone speech. *IEEE Transactions on Speech and Audio Processing*, 4(1):31.

Zissman, M. A. and Berkling, K. M. (2001). Automatic language identification. *Speech Communication*, 35(1):115–124.

Ziv, J. and Lempel, A. (1977). A universal algorithm for sequential data compression. *IEEE Transactions on information theory*, 23(3):337–343.